UNIVERSITY OF ALBERTA

AUTOMATIC RESONATOR ALIGNMENT AND POWER STABILIZATION OF HIGH POWER CO2 LASERS

BY

DALE RICHARD AKITT

©

A thesis submitted to the Faculty of Graduate Studies and Research in partial
fulfillment of the requirements for the degree of Doctor of Philosophy.

Department of Electrical Engineering

Edmonton, Alberta

Fall 1992

National Library
of Canada

Bibliothèque nationale
du Canada

Canadian Theses Service

Service des thèses canadiennes

Ottawa, Canada
K1A 0N4

The author has granted an irrevocable non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of his/her thesis by any means and in any form or format, making this thesis available to interested persons.

The author retains ownership of the copyright in his/her thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without his/her permission.

L'auteur a accordé une licence irrévocable et non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de sa thèse de quelque manière et sous quelque forme que ce soit pour mettre des exemplaires de cette thèse à la disposition des personnes intéressées.

L'auteur conserve la propriété du droit d'auteur qui protège sa thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

ISBN 0-315-77217-4

Canada

UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR: Dale Richard Akitt
TITLE OF THESIS: Automatic Resonator Alignment and Power Stabilization of High Power CO2 Lasers
DEGREE: Doctor of Philosophy
YEAR THIS DEGREE GRANTED: 1992

Permission is hereby granted to the University of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as herinbefore provided neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.

D R Akitt

5 Sheridan Drive
St. Albert, Alberta,
Canada, T8N 0J1

Date Aug 9/92

UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled AUTOMATIC RESONATOR ALIGNMENT AND POWER STABILIZATION OF HIGH POWER CO2 LASERS.

H Seguin
Supervisor: Prof. H. J. J. Seguin

D Dale
Committee Member: Prof. D. Dale

Keith A. Stromsmoe
Committee Member: Prof. K. Stromsmoe

F Vaneldik
Committee Member: Prof. F. Vaneldik

John C. Salmon
Committee Member: Prof. J. Salmon

Dr M. E. Pedinoff
External Examiner: Dr. M. E. Pedinoff

Date July 17/92

DEDICATION

To the love of my life Emily, without whose love and
encouragement this work could not have been completed

ABSTRACT

The high power carbon dioxide laser has become a very important tool in the area of industrial materials processing. This is due primarily to the enormous optical intensities which the device can attain. Unfortunately, it has proved increasingly difficult to construct machines capable of maintaining a consistently uniform output beam at elevated power levels, for extended periods. These difficulties are often attributed to thermal distortion of the resonator optics. Because of this fact, researchers have invested considerable effort in the construction of water cooled, thermally stable optical benches, upon which to mount the resonator components. This approach is very expensive, and has provided only a partial solution to the problem.

A different approach, and the subject of this thesis, is to construct a feedback control system which actively re-positions the optical elements, so as to maintain optimum resonator alignment. Preliminary experiments demonstrated that this concept was indeed feasible. In this first study, a circular array of small thermistors was utilized as the feedback element. Indicators of mode uniformity in the vertical and horizontal directions were extracted from the array using electronic techniques. A relay type controller monitored these error measures and subsequently activated a pair of motor-micrometers fitted to one of the primary resonator mirrors. Although this thermistor-based system demonstrated the viability of the concept, the sensor array's slow response limited its performance. Subsequent experiments utilizing high speed pyroelectric detectors in place of the thermistors solved the problem.

The next series of experiments was dedicated to finding a method of simultaneously aligning both primary resonator mirrors. This was accomplished by configuring the sensor array to monitor not only beam uniformity, but also beam position. A new controller was then developed which adjusted one mirror based on the position of the output beam, while the second mirror was employed to continuously maintain mode uniformity. This "two mirror" control system has demonstrated the ability to consistently maintain optimum resonator alignment for indefinite periods, at arbitrary power levels up to 10 kW.

ACKNOWLEDGMENTS

The author would like to express his sincere appreciation to the many individuals involved in the research project represented by this thesis. A special note of gratitude goes to the supervisor, Dr. H. J. J. Seguin. His style of leadership created an atmosphere which helped make this undertaking an exceptional growth experience.

In addition, many thanks are extended to the members of the laser research team for their invaluable technical support. In particular, Hugo Reshef and Chris Sellathamby provided indispensable aid with all stages of the research project. The author would also like to acknowledge the excellent services provided by all of the contributing individuals working in the laser laboratory, machine shop, and optics laboratory.

TABLE OF CONTENTS

Chapter **Page**

1. INTRODUCTION 1
   - 1.1 Unstable Resonator Systems 1
   - 1.2 Feedback Control of Laser Resonator Alignment 3
   - 1.3 Project Overview 4

2. LOW POWER SIMULATION 6
   - 2.1 Experimental Apparatus 6
   - 2.2 Experimental Procedure 9
   - 2.3 Experimental Results 10
   - 2.4 Discussion and Conclusions 10

3. THERMISTOR BASED SYSTEM FOR 5 kW PIE-3 LASER 12
   - 3.1 Beam Profile Sensor Design 12
   - 3.2 Servo Design 14
   - 3.3 Test Laser 15
   - 3.4 Controller Design 18
   - 3.5 Experimental Results 20
   - 3.6 Discussion and Conclusions 22

4. PYROELECTRIC BASED SYSTEM FOR 5 kW PIE-3 LASER 24
   - 4.1 Sensor Considerations 25
   - 4.2 Pyroelectric Detector Operation 26
   - 4.3 Sensor Array 27
   - 4.4 Sensor Interface Electronics 29
   - 4.5 Experimental Setup 31
   - 4.6 System Model 32
   - 4.7 Parameter Estimation 35
     - 4.7.1 Alignment System Parameters 35
     - 4.7.2 Power Control System Parameters 36
   - 4.8 Controller Design 40
   - 4.9 Experimental Results 42
     - 4.9.1 Alignment Controller Tuning 42
     - 4.9.2 Power Controller Tuning 42
     - 4.9.3 Transient Response 43
     - 4.9.4 Steady-State Response 43
   - 4.10 Discussion 47
     - 4.10.1 Performance 47
     - 4.10.2 Stability 47
   - 4.11 Summary 48

5. FEEDBACK CONTROL OF BEAM POINTING ANGLE.................. 50
  5.1 Beam Steering in Optical Resonators.................. 51
  5.2 Sensor Interface Circuit Modification.................. 52
  5.3 Extended Plant Model.................. 53
    5.3.1 Parameter Estimation.................. 55
  5.4 Controller Design.................. 57
  5.5 Experimental Results.................. 58
    5.5.1 Controller Tuning.................. 58
    5.5.2 Transient Response.................. 58
    5.5.3 Steady-State Response.................. 60
  5.6 Discussion.................. 61
    5.6.1 Position Controller Performance.................. 61
  5.7 Summary.................. 63

6. PROTOTYPE CONTROLLER DESIGN.................. 64
  6.1 Test Laser.................. 65
    6.1.1 Optical System.................. 65
    6.1.2 Power Supply.................. 66
  6.2 Hardwired Controller Installation.................. 66
    6.2.1 System Overview.................. 66
    6.2.2 Alignment Servo Installation.................. 68
    6.2.3 Sensor Array Installation.................. 68
    6.2.4 Controller Circuit Boards.................. 69
  6.3 Microcomputer Installation.................. 71
    6.3.1 Hardware.................. 71
    6.3.2 Software.................. 71
  6.4 System Model.................. 73
    6.4.1 Alignment System Parameter Estimation.................. 76
    6.4.2 Power Control System Parameter Estimation.................. 80
  6.5 Controller Design.................. 83
    6.5.1 Control Program Structure.................. 83
    6.5.2 Proportional-Integral Power Controller.................. 85
    6.5.3 Uniformity and Position Controller.................. 86
  6.6 Experimental Results.................. 87
    6.6.1 Power Controller.................. 87
    6.6.2 Alignment Controller.................. 87
    6.6.3 Steady-State Performance.................. 91
  6.7 Discussion.................. 92
  6.8 Summary.................. 93

7. SUMMARY AND RECOMMENDATIONS.................. 95
  7.1 Summary.................. 95
  7.2 Recommendations for Future Research.................. 97

REFERENCES 99

APPENDIX I - DIGITAL FILTER DESIGN 103
  1. First Order Lowpass Filter 103
  2. Third order Butterworth filter 104

APPENDIX II - TUTSIM SIMULATIONS 105
  1. Alignment System Simulation 105
  2. Power System Simulation 105
  3. Alignment Simulation Model Listing 110
  4. Power Control Simulation Model Listing 114

APPENDIX III - SCHEMATIC DRAWINGS 116

APPENDIX IV - SOFTWARE LISTINGS 158
  CONTROL.C 159
  CONTROL.H 174
  CONTROL.PAR 178
  SUBS.C 179
  DISPLAY.H 188
  P40.C 196
  P40.H 198
  MAKEFILE 199

APPENDIX V - OPERATING MANUAL FOR CONTROL/DATA-ACQUISITION SYSTEM 199
  I. Graphical Interface Operation 200
    Data Collection 203
    Parameter Modification 203
    Motor-Micrometer Speed Adjustment 203
  II. Operation of the Hardwired Controller 204
    Alignment of the Sensor Array 204

LIST OF ILLUSTRATIONS

Figure **Page**

1.1 Basic Alignment Control System 3
2.1 Low Power Simulation Apparatus 7
2.2 Rotating Wand Sampling Apparatus 8
2.3 Uniformity Measure Derivation 9
2.4 Low Power Simulation System Model 10
2.5 Response of Low Power Automatic Alignment System 11
3.1 Thermistor Test Apparatus 13
3.2 Thermistor Response to Infrared Radiation 13
3.3 Thermistor Array Circuit Board 14
3.4 Alignment Servo for Multikilowatt Laser 14
3.5 PIE-3 5 kW CO2 Laser 15
3.6 Folded Path Laser Resonator 16
3.7 Change in Thermistor Resistances when exposed to 1% Sample of 4 kW Beam (% change from cold value) 17
3.8 Quadrant Summing Circuit 18
3.9 Block Diagram of High Power Laser Mode Control System 19
4.1 Response of Pyroelectric Detector 26
4.2 Pyroelectric Detector Equivalent Model 26
4.3 Pyroelectric Detector Array Layout 28
4.4 Operation of Interface Circuits 29
4.5 Peak Detector Circuit 30
4.6 Experimental Setup 32

4.7 Laser Alignment and Power Control System 34

4.8 Estimation of Alignment System Parameters 36

4.9 Variac Estimation Procedure 37

4.10 Power Control System Estimation Results 39

4.11 Laser Alignment and Power Control System Model with Estimated Parameters 40

4.12 Non-linear Controller Structure 41

4.13 Transient Response of Closed-Loop System 44

4.14 Long-Term Open-Loop Performance 45

4.15 Long-Term Closed-Loop Performance 46

5.1 Beam Steering Effect in a Laser Resonator 51

5.2 Sensor Grouping for Position Detection 53

5.3 System Model for Two Mirror Control 54

5.4 Estimation of Position Measure Sensitivity Factors 56

5.5 Completed PIE-3 Laser System Model 57

5.6 Beam Position Relay Controller 58

5.7 Transient Response of Beam Position Controller 59

5.8 PIE-3 Steady-State Performance with Power and Uniformity Control but without Position Control 60

5.9 PIE-3 Steady-State Performance with Power, Uniformity and Position Control 61

6.1 Folded Path Laser Resonator 65

6.2 Block Diagram of the PIE-4 Mode Controller Installation 67

6.3 PIE-4 Beam Profile Sensor Array 69

6.4 Arrangement of the Mode Controller Circuit Boards 70

6.5 Data Acquisition System Graphical Display 72

6.6 Mathematic Model for the PIE-4 Mode Control System 74

6.7 TUTSIM Model used to Estimate the Uniformity System Parameters .. 76

6.8 Estimation of the Vertical Uniformity and Backlash 77

6.9 Estimation of the Horizontal Uniformity and Backlash 78

6.10 Estimation of the Position Measures 79

6.11 Estimation of the PIE-4 Power Control System 81

6.12 Completed PIE-4 System Model 82

6.13 The Data-Acquisition and Control Program 84

6.14 Power Controller Step Response 88

6.15 Performance of the Alignment Controller in Recovering from a Severe Misalignment of both Resonator Mirrors 90

6.16 Steady-State Performance of the PIE-4 Mode Controller 91

LIST OF SYMBOLS

P	Power
I	Electrical Current
V	Electrical Voltage
R	Electrical Resistance
C	Capacitance
R_i	Responsivity of Pyroelectric Crystal
Γ	Pyroelectric Crystal Material Parameter
d	distance
t	time
τ	time constant
T_s	sampling period
K	gain
n	sample number
N	total number of samples in the data vector
Θ	angle
SCR	Silicon Controlled Rectifier
FET	Field Effect Transistor
PIE	Photo-Initiated, Impulse-Enhanced, Electrically-Excited
PI	Proportional-Integral
DC	Direct Current
AC	Alternating Current
ASCII	American Standards Code for Information Interchange
TUTSIM	Twente University of Technology Simulation Program
λ	Wavelength
RAM	Random Access Memory

CHAPTER 1

INTRODUCTION

The CO_2 laser was first demonstrated by Patel in 1964.[1] Since that time, many improvements have been made to the device, resulting in its application to a variety of materials processing tasks in industry.[2-4] Some of these applications include welding, surface hardening, heat treating of metals, and surface alloying, etc. Many companies are now manufacturing commercial lasers for use in such applications. However, many of these lasers are operating at below the 1 kW. output level. The increasing need to perform efficient processing of much larger workpieces in situations such as thick section welding and cutting, has resulted in the development of devices capable of optical output in the multikilowatt range. Most lasers operable at these elevated power levels are of the transverse flow, gas transport geometry.[5] Lasers with output powers up to 25 kW have been constructed using this design concept.[6]

Transfer of these devices from the research laboratory to industry has been hindered by the quality of the output beam generated by them. In any laser device, optical element quality, and resonator alignment stability have a pronounced influence upon the uniformity and effectiveness of the output beam. In this context, favorable results in materials processing applications invariably require an output beam which consistently exhibits uniform power distribution.[7] A mode which is non-uniform is impossible to focus to the small diameters that are essential for high performance cutting and welding. Unfortunately, experience has shown that with presently utilized laser technology it is difficult to maintain a good quality beam at elevated power levels for extended periods of time, without operator intervention.

1.1 Unstable Resonator Systems

The resonators used in multikilowatt CO_2 lasers are almost exclusively of the unstable variety. Traditional resonators using a partially transmitting element at one end of the laser cavity are not viable at higher power levels. Unstable resonators however, offer the unique capability to extract energy from the resonator cavity using only reflective optics.[8] This allows the optical components to be constructed from metals such as aluminum or molybdenum. Water cooling can then be utilized to carry away excess heat.

Thermal loading is often a problem in these machines, since it affects not only the resonator optics, but the entire laser vessel. As the temperature of the laser gas mixture progresses from a cold startup condition to operating temperature, the containment vessel heats, and expands. Such a situation can be disastrous to resonator alignment, especially in systems where the optical system is fastened directly to the enclosure walls. Thus far, attempts to remedy these problems have focussed primarily on the use of thermally stabilized optical benches.[9] These systems employ water cooling to stabilize the temperature of an optical bench upon which the optical components are mounted. While the technique appears reasonable, the additional materials and equipment required to perform the necessary water circulation and temperature control, result in a prohibitively expensive and mechanically complicated solution. Furthermore, although this technique does provide improved performance, drifts in mode uniformity are not eliminated and so periodic adjustment of the resonators alignment is still required.

Unfortunately, thermally induced mechanical distortion is not the only adversary of resonator alignment stability. Vibration, wear and tear on the optical positioning components, and other effects, all tend to make optimum resonator alignment an ongoing issue. To further complicate matters, depending on the power level at which the device is operated, the refractive index of the gas medium varies across the discharge volume. This causes optical rays travelling through the discharge to refract by differing amounts, depending on their position within the gain medium. At very high power levels this effect is sufficient to induce an apparent resonator misalignment, even though the resonator remains physically in a perfectly aligned condition. Presently, the only method available for dealing with this effect is adjustment of the optical resonator's alignment.

Recently, researchers have begun using sophisticated monitoring systems to accurately examine output beam quality; a skilled human operator is then called upon to analyze this information and make appropriate alignment corrections to maintain beam quality.[10-19] An alternative approach, and the subject of this research project, is the application of feedback control techniques to achieve a system which continuously monitors the quality of the lasers output mode, and then actively adjusts the positioning of the resonator optical components to maintain optimum alignment. Previous work by many researchers with other types of mechanical and electrical systems, has demonstrated that the performance of inherently unstable mechanisms can often be significantly improved by the incorporation of a feedback control loop.

1.2 Feedback Control of Laser Resonator Alignment

An automatic alignment system is shown in its simplest form in Fig. 1.1. The first requirement is a servo which is capable of aligning the optical components in response to an electrical control signal. This can be accomplished by replacing the manual micrometers normally used in optics alignment systems, with motor micrometers. Secondly, the system needs a sensor which can provide a measure of the laser beam's cross sectional uniformity with sufficient accuracy to allow detection of any significant degradation in beam quality. This device must obtain information describing the mode quality without interfering with the laser's output. And lastly, a method of analyzing the information obtained from this sensor is required to direct the alignment adjustments which will provide the best improvement in beam quality.

The feedback control system approach introduced here promises several

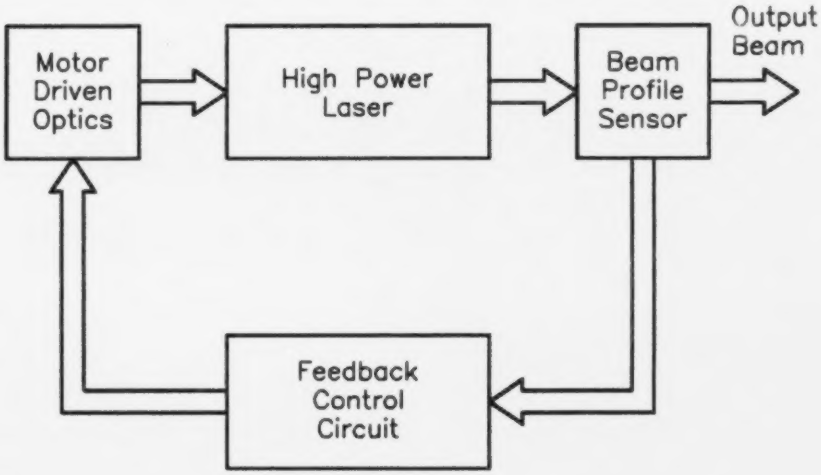


Fig. 1.1 Basic Alignment Control System

advantages over current techniques of maintaining laser beam stability. The most obvious benefit would be elimination of the very expensive optical benches presently employed in these lasers. This type of system would contribute to the feasibility of unsupervised operation of these lasers, a significant benefit in a manufacturing environment. Most importantly, from an industrial viewpoint, improved quality control would result, which in turn would lead to fewer rejects and subsequent increases in productivity.

1.3 Project Overview

Before beginning experiments on an actual high power laser, a project was initiated to simulate the behavior of an unstable resonator system at very low power. This experiment provided a simple test of the feedback control concept, without the complication and expense of high power laser operation. A 35 W CO_2 laser, together with a system of mirrors, was configured to obtain an output similar to that produced by a laser equipped with an unstable resonator. A commercial pyroelectric quadrant detector, was used to provide a measure of beam nonuniformity. A simple control system was subsequently designed to utilize the information provided by this sensor, so as to maintain proper alignment of the mirror system. The effectiveness of this system was evaluated qualitatively by observing how well the system returned the mirror to proper alignment, after a deliberate misalignment was introduced. Time response of the system was also documented in the form of plots of system error signal vs. time. The results obtained from these experiments were sufficiently encouraging to motivate further work with a 5 kW PIE laser.

While this low power simulation provided valuable insight into the control problem, it was expected that the behaviour of an actual resonator system would be quite different, due to the presence of an active gain medium between the resonator mirrors. The next step then was to install a beam profile sensor and control servos on the 5 kW PIE-3 laser, a machine available here in the University of Alberta laser laboratory. Unfortunately, during the course of the low power experiments, the fragile (and expensive) pyroelectric quadrant detector was damaged beyond repair. It was apparent that this device was too delicate for use in industrial applications, so a new sensor was designed based on an array of small thermistors. A controller was subsequently designed which utilized information provided by this sensor array to maintain proper alignment of a single resonator mirror. It was shown that the system could properly maintain resonator alignment during extended periods of laser operation. However, the experiments also revealed that the thermistor elements responded so slowly that the system provided inadequate transient response during startup periods.

The next undertaking then, was the design and construction of a much faster sensor array composed of small, individual pyroelectric detectors. These small pyroelectric elements were found to be far more durable, and much less expensive, than the original single crystal quadrant detector. This new sensor array was configured to supply information which allowed the alignment of both resonator mirrors. A special interface circuit was developed to convert the pulsed signals provided by these elements into a form usable by the controller circuitry. At the same time a microcomputer-based data-acquisition system was installed to provide for accurate data collection. This data collection system was employed to gather data which was subsequently used to construct a "control system model" for the assembly. The controller circuitry was then expanded to accomplish simultaneous alignment of both resonator mirrors. Tests with the new sensor and controller at this point demonstrated the ability of the system to automatically maintain complete resonator alignment for indefinite periods of time. Significantly improved transient response was also observed.

To this point in the experiments, system electronics were constructed entirely on breadboards, a less than ideal situation from the standpoint of electrical noise pickup and mechanical stability. Consequently, the next project was the design of a prototype controller, suitable for installation on the 25 kW PIE-4 laser. Printed circuit boards were designed to accomodate all of the electronics interface and control circuits. Along the way, various improvements were made to the electronic circuits and control algorithms. In addition, software was written to allow on-line, graphical display of information obtained from the sensor array and control circuitry. This software proved to be invaluable for performance evaluation and tuning of the new controller. It also extended the usefulness of the sensor array to its utilization as a diagnostic tool, in addition to its function as a feedback control element. Performance of this prototype system was tested at power levels up to 10 kW using the PIE-4 laser.

CHAPTER 2

LOW POWER SIMULATION

Before investing considerable time and expense in experimenting with a high power laser system, a low power simulation was performed. This approach allowed evaluation of the feedback control concept at relatively low cost, and without risk to any high power laser equipment. The experiment provided a convenient opportunity to test a method of measuring mode uniformity, and in general, contributed to a better understanding of unstable resonator behavior.

A pair of aluminum mirrors was used to construct a benchtop apparatus which could roughly imitate the behavior of an annular output resonator, similar to those commonly used in multikilowatt lasers.[20] Laser energy for the system was provided by a commercial 35 Watt unit. A pyroelectric quadrant detector, also available commercially, was employed to measure the laser beam's uniformity. A simple feedback control system was then designed to maintain proper alignment of the mirror system. The purpose of this chapter is to provide a description of the experimental apparatus used in this simulation and to present basic data demonstrating the viability of a closed loop alignment system. Results obtained here led eventually to experiments with the 5 kW PIE-3 laser.

2.1 Experimental Apparatus

A diagram of the experimental setup is shown in Fig. 2.1. The individual components of this apparatus are described below:

1. Mirror Assembly - As discussed earlier, the optical systems utilized in high power lasers are usually unstable resonators. To obtain similar behaviour, the test system of Fig. 2.1 was assembled using two small aluminum mirrors. A small hole in the center of the concave mirror allows the 10.6 μm wavelength beam from the injection laser to enter the resonator cavity, where it expands on each round trip until the beam overflows the outer edge of the convex mirror.[21] Each mirror was secured to a double axis, gimbal type mount.

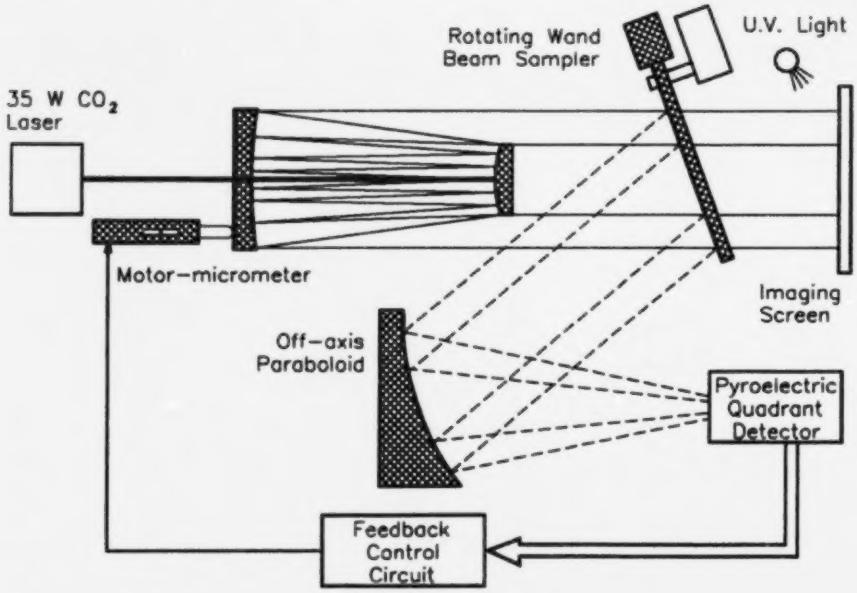


Fig. 2.1 Low Power Simulation Apparatus

Motor-micrometers on the concave mirror mount provided alignment of one mirror under electronic control.[1]

2. Motorized micrometers - The motor-micrometers used to align the concave mirror were commercially available units. The companion controller units were conveniently equipped with an operational feature allowing bidirectional movement, proportional to the input voltage. Servo speed and direction were determined by input signal polarity and magnitude, respectively.[22]

3. Sampling Apparatus - Sampling of high power laser beams is difficult to perform using partially reflecting optics because of the high intensity levels involved. Instead, a more common method employs a small pie shaped metal mirror like the one illustrated in Fig. 2.2. This specially designed mirror is mounted on a small electric motor, and then rotated across the path of the output beam at an appropriate angle, thereby sampling the complete cross-

---

[1] To preserve clarity, fig. 2.1 shows only a single micrometer on the concave mirror mount.

section of the laser beam profile on each rotation. The percentage of energy coupled out of the main beam is determined by the subtended angle Θ, identified in Fig. 2.2. The average power coupled out by the rotating wand is described by:

$$\frac{P_o}{P_i} = \frac{\Theta}{360} \qquad (2.1)$$

For these experiments Θ was fixed at 3.6°; thereby delivering 1% of the main beam to the pyroelectric detector. The wand was constructed of an aluminum alloy; the optical surface finishing was performed in-house on a micro-surface lathe.

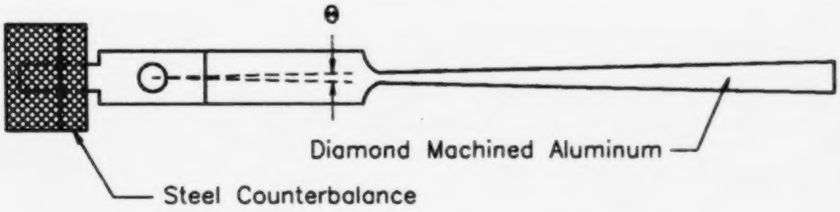


Fig. 2.2 Rotating Wand Sampling Apparatus

4. Off-axis Paraboloidal Mirror - Pyroelectric detectors are available only with very limited dimensions of detector surface area. This is because of the difficulty involved in growing single crystals of lithium tantalate in large sizes. The particular detector used here provided a 2 cm. x 2 cm. detector surface. It was therefore necessary to reduce the diameter of the annular beam. This was performed with the aid of an off-axis paraboloidal mirror.[23] The focal length and turning angle were 7.0" and 40°, respectively.[2] This element was also produced in-house.

5. Pyroelectric Quadrant Detector - The detector used as a beam profile sensor was a commercially available pyroelectric quadrant detector.[24] This device consists of four electronically separate detectors, arranged in a square matrix on

---

[2] The turning angle is measured between the incident and reflected beam. (see experimental apparatus diagram)

a single 2 cm. x 2 cm. lithium tantalate crystal. The unit is equipped with internal integrated electronics to perform the necessary impedance conversion, and to provide signal amplification.

6. Imaging Screen - This device is a commercially available screen, marketed for observation of infrared laser beams. When illuminated with ultraviolet light, the screen fluoresces with a bright yellow color. Simultaneous heating by an infrared source quenches this fluorescent effect, causing the screen to darken in proportion to the intensity of laser energy received. As such, the unit provided a convenient means for observing the effect of alignment changes upon the output beam's uniformity.[25]

2.2 Experimental Procedure

Fig. 2.3 illustrates how the pyroelectric detector was utilized to derive a measure of the annular beam's nonuniformity. By comparing the output signal intensities from the top and bottom halves of the detector, a measure of uniformity mismatch in the vertical direction was obtained. Similarly, a uniformity error measure for the horizontal direction was calculated by comparing the left and right halves. In equation form:

$$\mathrm{YUNIFORMITY} = (A+B) - (C+D) \quad (2.2)$$
$$\mathrm{XUNIFORMITY} = (A+D) - (B+C) \quad (2.3)$$

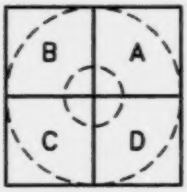


Fig. 2.3 Uniformity Measure Derivation

By defining the above uniformity measures as outputs, and neglecting interaction between the two alignment axes, the simple model of Fig. 2.4 was constructed. It was assumed that the plant transfer functions, G_{py} and G_{px}, reduced to simple constants[3]. After tuning the feedback controller gains, K_{cy} and K_{cx}, a chart recorder was used to plot the system's performance in returning the mirror to proper alignment after a suitable misalignment was introduced.

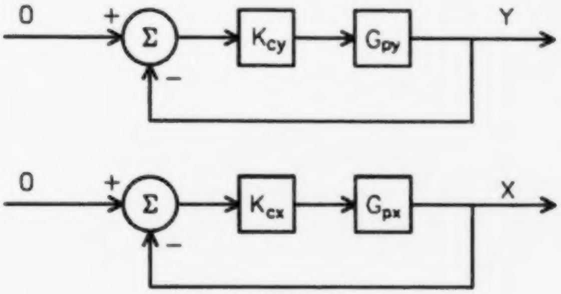


Fig. 2.4 Low Power Simulation System Model

2.3 Experimental Results

The controller response along the vertical axis is plotted in Fig. 2.5 for three values of gain K_c. Virtually identical results were observed for the horizontal direction. For each response curve in Fig. 2.5, the concave mirror was misaligned to the point where only about half of the annular mode remained. This required a micrometer displacement of approximately 50.0 μm from the aligned position. The controller was then returned to automatic and the response was recorded using a chart recorder.

2.4 Discussion and Conclusions

Fig. 2.5 reveals a response typical of a 2nd order system, equipped with a proportional feedback controller. As illustrated in this figure, a maximum time of approximately 4 seconds was required to return the mirror to proper alignment. It was found that the system could achieve proper alignment from virtually any

---

[3] Simple observation revealed that the assumptions of a linear plant and no interaction between alignment axes were reasonable. At this point in the experiments it sufficed to assume this simplified plant and simply tune the controller by trial and error.

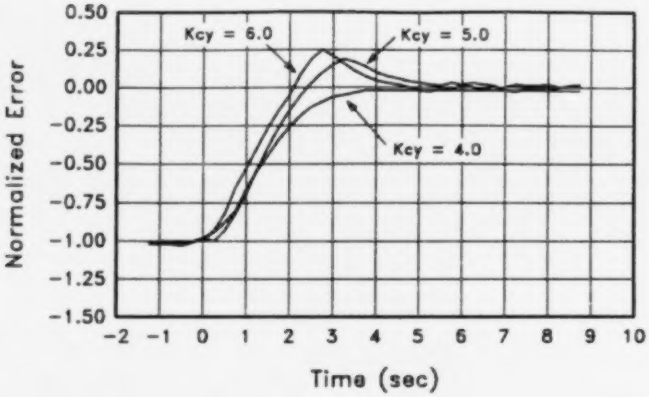


Fig. 2.5 Response of Low Power Automatic Alignment System

extreme of misalignment within about 5 seconds. Of course, the system could only be expected to acquire proper alignment if initial alignment conditions were such that enough laser energy was present to drive the controller in the proper direction. This condition is satisfied under normal operating conditions, when the controller is continuously maintaining proper alignment.

The results provided by this simple controller were sufficiently encouraging to prompt further experimentation with the 5 kW PIE-3 laser. Unfortunately during the course of these experiments the lithium tantalate crystal, the heart of the pyroelectric quadrant detector, was cracked. As will be discussed in the next section, this required the development of a new beam profile sensor before experiments could begin with the high power laser.

CHAPTER 3

THERMISTOR BASED SYSTEM FOR 5 kW PIE-3 LASER

The low power simulation described in Chapter 2 convincingly demonstrated the feasibility of automatic resonator alignment. The next undertaking, then, was the development of a system which could be installed on the PIE-3, 5 kW, CO_2 laser. As mentioned previously, the pyroelectric quadrant detector was damaged during the low power experiments. This raised concerns regarding its suitability as a beam profile sensor. The quadrant detector was a very expensive device and as was clearly shown, it was very susceptible to mechanical damage. Consequently, such a fragile component would not be reliable in an industrial setting. Therefore, a new beam profile sensor was designed for installation on the high power laser. In addition to a more rugged construction, the new design reduced the cost of the feedback sensor from $6000 to approximately $400.

Concurrent with the construction of the new sensor, servos to accomplish adjustment of alignment and power input on the PIE-3 laser were developed. An electronic feedback controller was then designed to stabilize both resonator alignment, and laser power output. Design details for the beam profile sensor, servos, and electronic control circuits are provided below. Experimental results verifying the system's ability to maintain resonator alignment and laser output power are also presented.

3.1 Beam Profile Sensor Design

After the pyroelectric quadrant detector was damaged, the idea of designing a new sensor composed of an array of small thermistors was conceived. Before constructing the array, a simple experiment was performed to determine the sensitivity of an individual thermistor. The apparatus of Fig. 3.1 was used to measure the response of a single element, when exposed to infrared radiation supplied by a small CO_2 laser. Resistance measurements were recorded for two randomly chosen thermistors, as laser power was varied between 0 and 180 mW.

Fig. 3.2 reveals a non-linear relationship, with thermistor resistance inversely dependent upon laser power. It can also be seen that a variation of only 180 mW of laser power is required to vary the thermistor resistance over the full useful range of the device.

The new beam profile sensor was constructed by mounting 48 of these thermistors on a printed circuit board in an annular array, as shown in Fig. 3.3. The dimensions of the array were selected to ensure that the entire sample of the annular beam fell within the inner and outer radii of the array. Ribbon cable connectors at each end of the circuit board provided electrical connection to individual elements in the array.

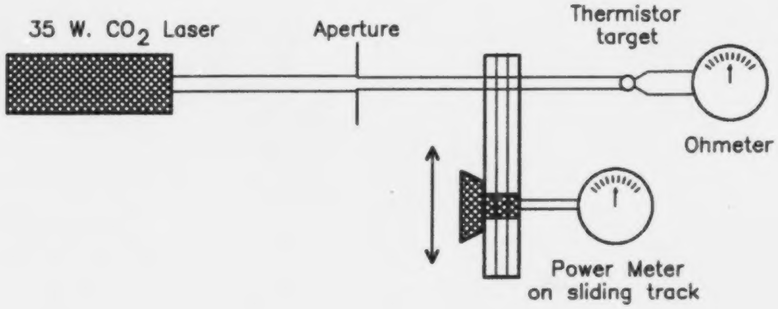


Fig. 3.1 Thermistor Test Apparatus

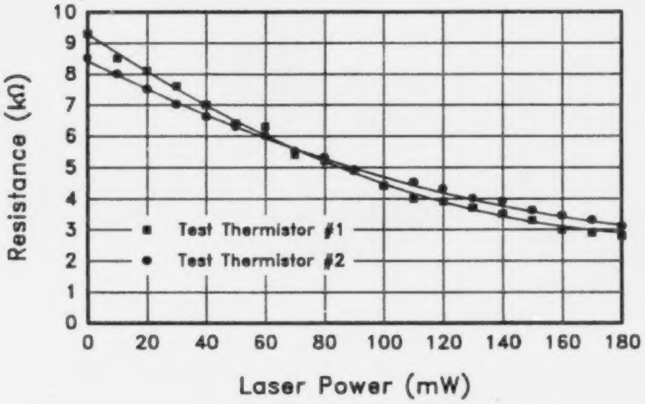


Fig. 3.2 Thermistor Response to Infrared Radiation

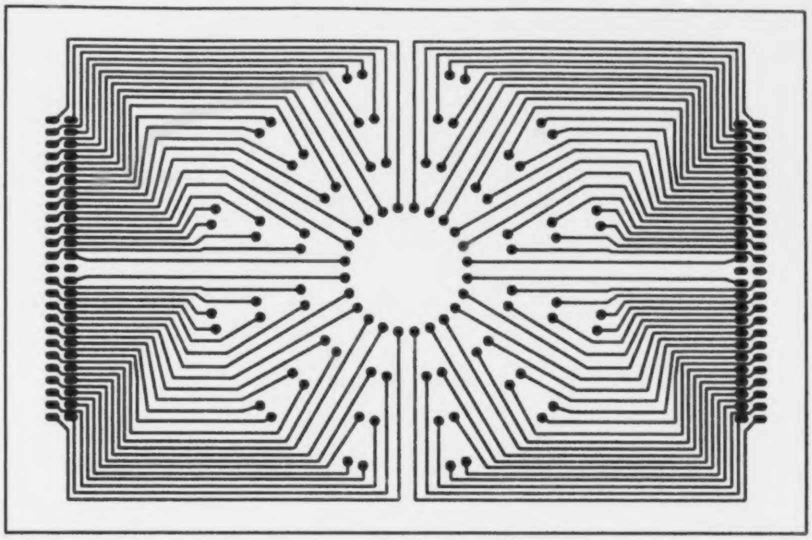


Fig. 3.3 Thermistor Array Circuit Board (scale 0.8:1)

3.2 Servo Design

Initial attempts to utilize motor-micrometers to perform alignment adjustments on the PIE-3 optical system gave poor results. The motor-micrometers used in the low-power simulation experiments provided insufficient torque to move the relatively heavy mirror mounting hardware present in the high power laser. To resolve this problem, the arrangement shown in Fig. 3.4 was used; this design takes advantage of the 485:1 gear reduction provided by the motor-micrometers to rotate

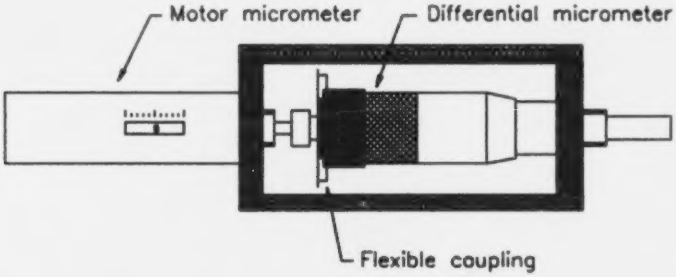
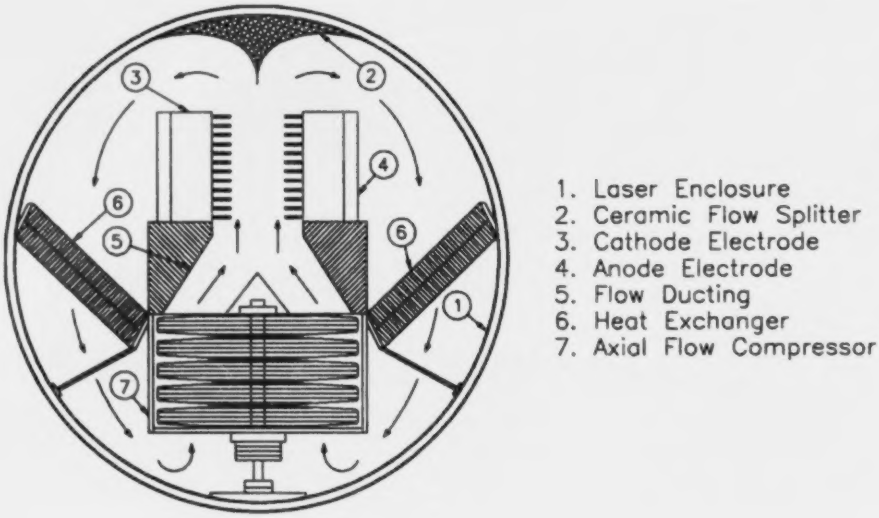


Fig. 3.4 Alignment Servo for PIE-3 Laser

the adjusting knobs on the manual differential micrometers. The motor-micrometers provided sufficient torque to drive the manual units, and the additional 40:1 gear reduction provided by the manual micrometers allowed alignment changes to be made at an appropriate rate, with excellent resolution.

3.3 Test Laser

Experiments were performed on an existing 5 kW **PIE**[1] CO_2 laser. Detailed descriptions of the individual portions of this laser system have been reported elsewhere, so only a brief review of the key components will be provided here.[26-28] The schematic diagram of Fig. 3.5 illustrates the main components of the PIE-3 laser. These include a 12-module pair of multi-element, fluid-ballasted, pin electrodes, 2 large heat exchangers, a linear array of 6 axial-flow compressors, and a folded-path optical resonator. Gas flow, discharge electric field, and optical axis were mutually orthogonal. Discharge length and inter-electrode spacing were 3 m and 90 mm, respectively. Identical multi-pin structures were used for both anode and cathode electrodes.



1. Laser Enclosure
2. Ceramic Flow Splitter
3. Cathode Electrode
4. Anode Electrode
5. Flow Ducting
6. Heat Exchanger
7. Axial Flow Compressor

Fig. 3.5 PIE-3 5 kW CO_2 Laser

---

[1] **PIE** is an acronym describing the patented discharge excitation process used in the test laser. It stands for **P**hoto-initiated, **I**mpulse-enhanced, **E**lectrically-excited.

The double-pass, water-cooled optical extraction system shown schematically in Fig. 3.6, consisted of two mirrors M1 and M2, with a 90 degree roof prism, M3, serving as the folding element. Output coupling through a ZnSe window was achieved using the skimmer, M4, and flat mirror, M5. Total resonator length was 860 cm, while the radii of curvature of mirrors M1 and M2 were -3500cm and 3800cm respectively. All optical components in the system were diamond machined from an aluminum alloy.

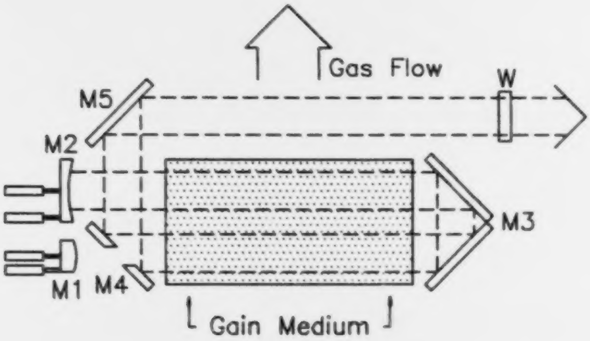


Fig. 3.6 Folded Path Laser Resonator

The new beam profile sensor was installed on the PIE-3 laser, along with a rotating wand designed to reflect a 1% sample of the main beam onto the sensor array.[2] The alignment servos were also installed on mirrors M1 and M2 at this time. A trial run was then performed to determine the operating point of the thermistors. The resistance variation of each element, from cold values, in response to the 1% beam sample, was measured at a power level of 4.0 kW, near the laser's maximum of 5 kW. Fig. 3.7 presents this data diagrammatically. Lines connecting the highest values of change were drawn to obtain a rough indication of the beam shape. The maximum observed change in thermistor resistance was 39%, indicating that the chosen sampling ratio of 1% was suitable to achieve proper sensor operation over the potential output power range of the laser.

---

[2] Since the thermistors exhibited a time-constant of approximately 4 seconds, the beam profile sensor array responded only to the average power of the sample provided by the rotating wand. The wand dimensions were selected to reflect 1% of the main beam's average power onto the sensor array. A fine metal screen was also placed in front of the sensor array; this helped compensate for the presence of localized hot-spots by refractive smoothing of the sample.

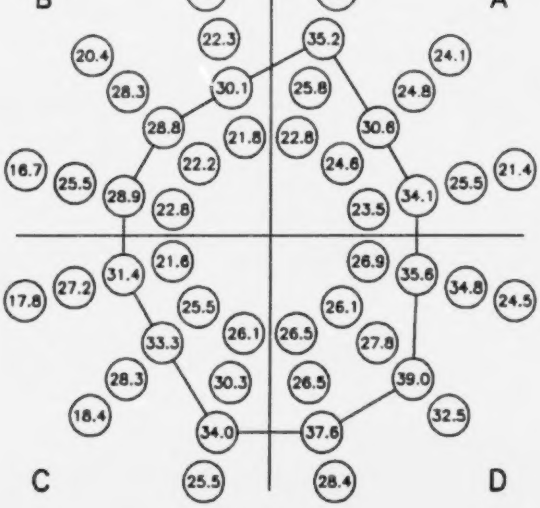


Fig. 3.7 Change in Thermistor Resistances
when exposed to 1% Sample of 4 kW Beam
(% deviation from cold resistance value)

In the low power simulation described earlier, the pyroelectric quadrant detector provided signals representative of the beam intensity in each quadrant. However, using an array of thermistors as the sensor required the design of interface circuits to convert the resistances of the individual thermistors into voltage levels, and to combine these voltage levels into quadrant sums which could be used to determine the degree of beam nonuniformity.

Thermistors from each quadrant were biased at 0.6 V and connected to the summing junction of a standard op-amp summing circuit, as portrayed in Fig. 3.8. In this manner 4 signals were obtained which indicated the average power present in each quadrant of the laser output.[29]

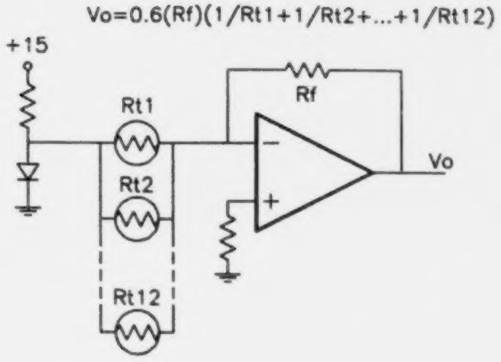


Fig. 3.8 Quadrant Summing Circuit

3.4 Controller Design

The design of a controller for the high power laser was complicated by the unavailability of powerful, variable speed motor-micrometers to perform the alignment adjustments. In spite of the micrometer modifications described earlier, it was found that traditional proportional control was not feasible. This was because depending on the load imposed on them, the micrometers would initiate movement at substantially differing magnitudes of actuating error signal; this unpredictable behavior proved to be very difficult to control. Manual experimentation revealed that superior performance was obtained in the jog mode, with the motor speed adjusted to a constant value. Consequently, the use of a non-linear, relay type controller was required.

Fig. 3.9 depicts the control strategy developed to utilize the information from the sensor array. The summer outputs, labeled A through D, were connected to additional summing circuits, which were configured to provide the voltage levels, A+B: B+C: A+D: and C+D. These signals were in turn fed to subtraction circuits, which generated the control signals, Y-ERROR and X-ERROR. Each of these signals furnished a measure of the beam's asymmetry in the vertical and horizontal directions, respectively. Level detector circuits were then used to examine these error signals, to determine if mode asymmetry was within pre-defined limits.[29]

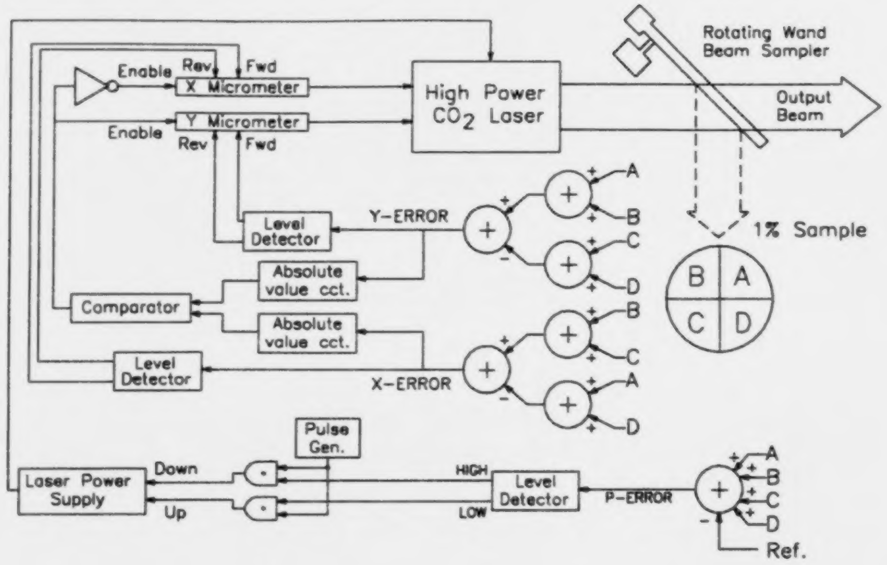


Fig. 3.9 Block Diagram of High Power Laser Mode Control System

The magnitude of these two error signals were simultaneously analyzed by comparator circuits. If the error in any of the 4 possible control directions exceeded a pre-set threshold, then a signal initiating mirror movement in the appropriate direction to reduce that error, was generated. The mirror adjusting motormics were driven sequentially such that the mirror axis exhibiting the largest error signal was enabled first. This control format, allowing mirror movement only along the axis having the larger error, compensated for interaction between the alignment axes.

In addition to the mirror positioning information outlined above, the thermistor sensor array also provided a convenient method for power stabilization. This feature was derived from the quadrant sum voltages, A+B+C+D. It is evident in Fig. 3.9, that this sum signal was directly proportional to the total average power output produced by the laser. An error signal was then obtained by subtracting a reference from this signal; appropriate adjustment of this reference allowed selection of the desired output power level.

The difference signal P-ERROR generated in this manner, was then compared to pre-defined limits, to determine if power level correction was required. If the error exceeded upper or lower thresholds, a motor-driven variac feeding the laser's DC power supply was automatically activated to correct that error. To allow time for the thermistor elements to respond to changes in output power level, the variac was driven with short pulses; a time period of about 15 seconds was provided between pulses.

3.5 Experimental Results

The PIE-3 laser used for evaluating the effectiveness of the mode and power controller outlined above was well suited for the task. The optical components, illustrated in Fig. 3.6, were hard-fastened to the laser tank doors. As a result, resonator alignment was continually subject to large temperature-induced excursions, because of distortion in the containment vessel upon discharge initiation. There were also a number of small leaks in the fluid ballasted electrodes, which contributed to an abnormally rapid deterioration of the lasing gas mixture. As a consequence of these factors, manual operator intervention had previously been required every few minutes, in order to preserve mode quality and power level.

The first measurements of system performance were taken to determine the power stability of the laser. In the 3 separate runs depicted in Fig. 3.10, identical operating conditions were maintained, except that the power adjusting control system was selectively enabled and disabled. In each case, the electronic mirror alignment system was left engaged. These plots clearly reveal how the laser's output power rapidly dropped off during normal, unsupervised operation. The graphs also document how that power level was restored and maintained to within about 3%, upon activation of the electronic control system.

In order to quantify the mode control aspects of the system, a quadrature procedure for data collection was employed; the rationale here being, that beam sampling was also performed on a quadrature basis. Measurements of the power level in each quadrant were made, first with no alignment changes during operation, and then with the automatic mirror adjusting system activated. The lower curves of Fig. 3.11 reveal how the laser's mode drifts if no alignment corrections are made to the resonator. However, when the feedback control loop is

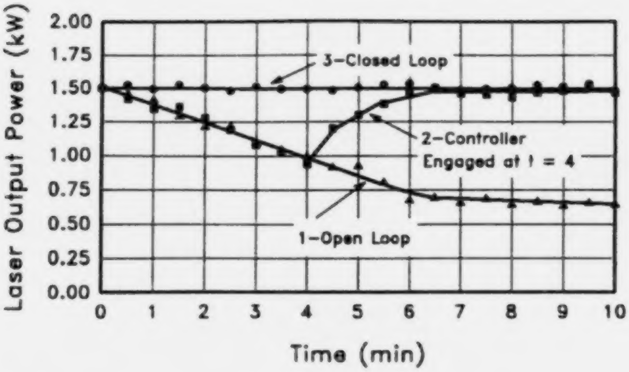


Fig. 3.10 Power Controller Performance

enabled the power output level in each quadrant is fully stabilized.

The observation from Fig. 3.11, that the absolute power levels measured in each quadrant were somewhat different, was found to be a constant characteristic of the degraded optical profiles present in the test laser. As such, equalization of the magnitudes of the integrated intensities within each quadrant of the laser's output beam was beyond the capability of either manual, or automatic control, when using these mirrors.

Fig. 3.10 also reveals the transient performance of the power controller. As can be seen, if the laser was started and run for some time without electronic power supervision, it would then take about 90 seconds for the system to reestablish the initial power level. Transient performance of the alignment portion of the controller was not documented because of difficulties imposed by the manual data collection methods employed at this stage of the project. However, since the alignment controller obtained information from the same sensor array, it was expected that the response time would be of the same order of magnitude as that exhibited by the power controller. Visual observations of the performance of the alignment controllers reinforced this conclusion.

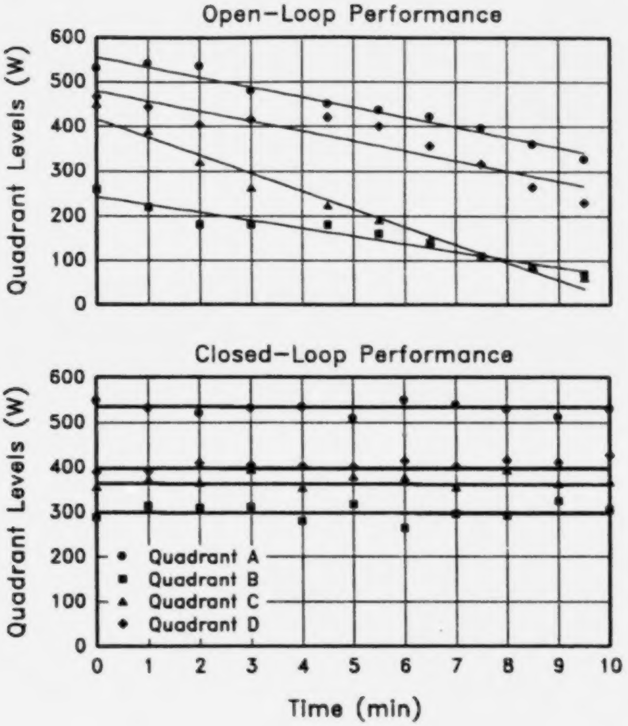


Fig. 3.11 Alignment Controller Performance

3.6 Discussion and Conclusions

Although the controller provided adequate steady-state performance, its relatively slow transient response would probably render the system ineffective under sporadic, and short on-off operational sequences of a laser. This excessively slow response time was attributed primarily to the large thermal time constant of the thermistors used in constructing the beam profile sensor. The devices used in these experiments have a thermal time constant of approximately 4 s., implying an equilibration time of 15-20 s. Response time of the DC variac, and alignment servos was negligible in comparison.

Another problem observed, was a slight downward drift in the level at which the power controller maintained output power, especially during extended runs. It was suspected that as the temperature of the sensor board rose, in response to heating by the beam sample, the thermistor elements were subjected to warm air currents which the controller perceived as an increase in beam intensity.

To alleviate these problems, work began on yet another sensor array, based on small, individual pyroelectric detectors. In addition to greater sensitivity and faster response, these devices, by nature of their operation, are not affected by low frequency drifts in ambient temperature. A control system such as outlined above, but substituting pyroelectric elements in place of the thermistors, should yield much improved performance.

CHAPTER 4

PYROELECTRIC BASED SYSTEM FOR 5 kW PIE-3 LASER

The first generation mode controller effectively demonstrated the improvement in laser performance which could be achieved by fitting a high power laser with an automatic alignment and power level controller. However, the system provided extremely poor transient response; this shortcoming would limit usefulness of the controller to longer term maintainance of alignment and power output.

Many newer applications in materials processing require rapid changes in laser power level during the processing task. Unfortunately, most existing high power lasers function effectively only when they are operated continuously. Consequently, these systems usually employ a beam dump for the laser's output, when it is not required for processing. This practise clearly compromises production efficiency, and subjects laser components to additional wear and tear. It would be considerably more efficient to equip the laser with a control system that is capable of quickly obtaining proper alignment and desired power level upon initiation of the processing task.

The poor transient performance demonstrated by the thermistor based controller was attributed to the thermal time constant of the individual sensor elements. Clearly, a faster feedback element was required. Improved performance was obtained by returning to the pyroelectric detector; this time however, an array of smaller and more durable individual detectors was utilized. To use this new sensor array, interface circuits were designed to convert the pulsed signals from the pyroelectric detectors into a form compatible with the feedback control electronics. The feedback control strategy was then modified to take advantage of the sensor's faster response.

As stated earlier, adequate monitoring of the systems response was impossible using manual data collection methods. To resolve this difficulty, a microcomputer-based, data-acquisition system was installed. This system was employed to collect data describing the plant's open-loop response; the resulting mathematical model was then utilized to program computer simulations. These simulations were very useful for evaluating possible controller structures, and for obtaining initial tuning parameters for the closed-loop system.

Details of the new sensor array, its interface electronics, and the data-acquisition system are supplied in the following sections. The methods used to obtain the open-loop model parameters are then described, followed by a discussion of the computer simulations and controller design. Finally, experimental results documenting the transient and steady-state responses of the new closed-loop system are presented.

4.1 Sensor Considerations

Several factors were considered in the selection of a suitable detector element to replace the thermistors in the beam profile sensor. The most important of these characteristics are: frequency response, sensitivity, linearity, stability, and physical size. Since photon energies are very low in the far infrared, room temperature sensing elements are limited to operation on a thermal basis. There are currently three commonly available detector types which fall into this category. These are thermistors, thin film thermopiles, and pyroelectric detectors.[30-32]

After consideration of the basic features of these detectors types, and the desired system performance, the pyroelectric detector was selected. A major factor in this choice was the device's fast response: in the hundreds of kHz. In addition, the pyroelectric effect requires that the incident infrared radiation be chopped. Consequently, a low average power sample of the laser's output, but encompassing full intensity of the main beam, can be obtained from a rotating wand, as in previous experiments. Another advantage of the pyroelectric detector is elimination of the offset problems induced by changing ambient temperature. This is a direct result of the poor low frequency response of the device. Finally, pyroelectric detectors provide an output voltage that is a linear function of the incident optical intensity.

4.2 Pyroelectric Detector Operation

The pyroelectric detector is constructed of materials such as lithium tantalate; these unique materials produce an electrical current as they experience a change in temperature.[32] Fig. 4.1 illustrates the response of a loaded pyroelectric detector to a step of infrared radiation. Note that the output occurs only during the initial moments after arrival of the infrared step; the output quickly decays to zero as the temperature of the pyroelectric material approaches steady state. This operational characteristic is the source of the pyroelectric detector's fast response. Fig. 4.2 depicts the simple circuit used to model the high frequency electrical behaviour of the pyroelectric detector, as it experiences a temperature change.

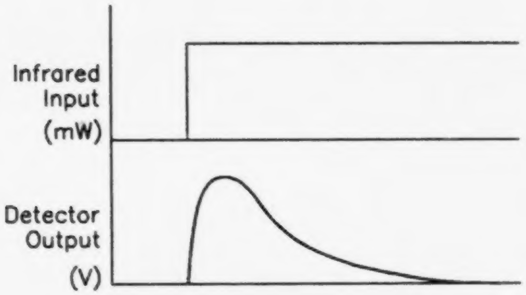


Fig. 4.1 Response of Pyroelectric Detector

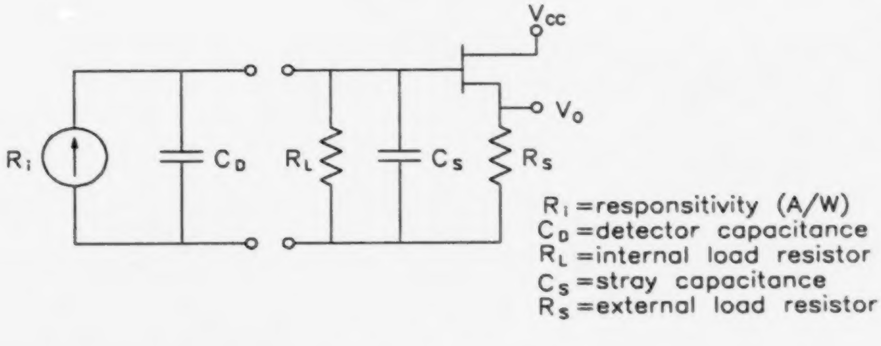

V_{cc}
V_o
R_i
C_D
R_L
C_S
R_S
R_i =responsitivity (A/W)
C_D =detector capacitance
R_L =internal load resistor
C_S =stray capacitance
R_S =external load resistor

Fig. 4.2 Pyroelectric Detector Equivalent Model

The diagram also includes the FET voltage follower which is often employed as a buffering stage. It can be seen that the detector is purely capacitive; this is by virtue of the high DC resistance of the pyroelectric material ($10^{13}\ \Omega$). Since this capacitance is fairly small (typically 30 pF), the detector exhibits a relatively high AC impedance. For most applications this impedance must be converted to a more practical low impedance output. Commercially manufactured pyroelectric detector's are available with integrated FET amplifiers, and a wide selection of internal load resistor values.

The current response of the pyroelectric element is independent of frequency of the source, once the time-rate of heat arrival is greater than the time-rate of material thermalization. The magnitude of current output is a linear function of the separation of the charge gathering electrodes, the material parameters, and at very low frequencies (< 1 Hz), the thermal time constant. Thus the detector current response (for other than low frequencies) is expressed as:

$$R_i = \frac{\Gamma}{d} \qquad (4.1)$$

where:

R_i = current responsivity (A/W)
d = electrode separation (cm)
Γ = material parameter ($5.49 * 10^{-9}\ \frac{\text{A cm}}{\text{W}}$ for lithium tantalate)

The overall voltage response of the detector plus the FET follower is dependent on the value of R_L (Fig. 4.2). The load resistor is selected to provide a tradeoff between output sensitivity and fast response. The risetime (10 to 90%), in response to a step, is determined by the electrical time constant:

$$t_r = \frac{\tau}{2.2} = \frac{R_L(C_D + C_S)}{2.2} \qquad (4.2)$$

4.3 Sensor Array

After considering the desired frequency response, the final detector selection was performed experimentally. The response of several units, each equipped with different window and load resistor combinations, was compared with the goal of achieving a satisfactory compromise between pulse response and sensitivity. A unit manufactured by Eltec Instruments, Inc. (model 406M34-1) was ultimately selected for use in the sensor array.[33] This detector was furnished with an internal load resistor and integrated FET buffer amplifier. The internal load resistor was taken as 10 MΩ; this value provides a 10 to 90% pulse rise time of 136 μs. A TO5 case with a germanium window (transmittance 48% at 10.6 μm) encloses each unit.

The sensor array, shown in Fig. 4.3, is comprised of 56 of these detectors, mounted on a printed circuit board in an annular array. Individual elements were spaced as tightly as possible, and sensor geometry was configured such that the entire cross-section of the sampled annular beam falls within the inner and outer radii of the array. Ribbon cable connectors at the edges of the circuit board provide access to individual elements in the array. The outputs from these connectors are fed to the interface circuit boards, which convert the pulsed output signals into DC form .

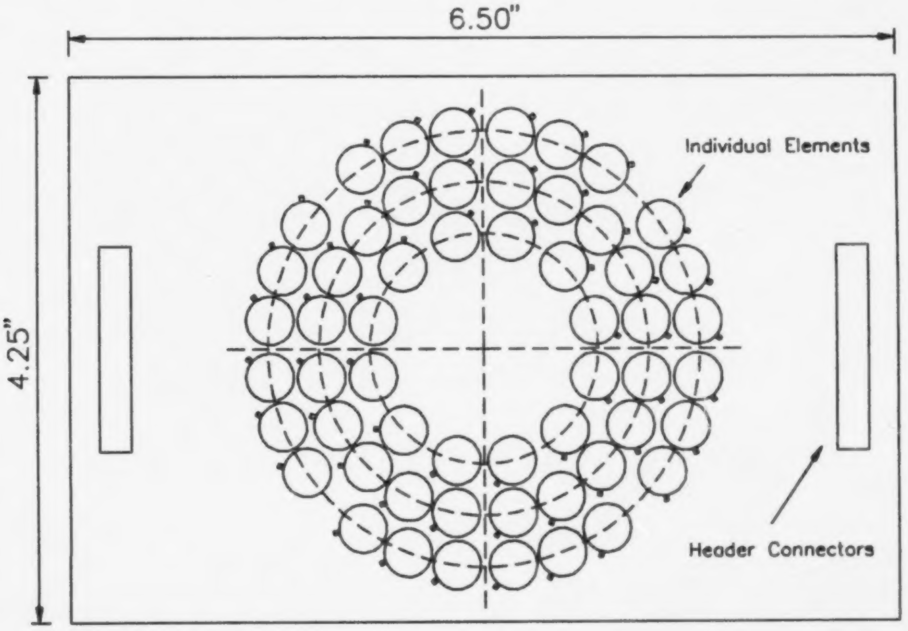


Fig. 4.3 Pyroelectric Detector Array Layout

4.4 Sensor Interface Electronics

The fast response provided by the pyroelectric detectors is obtained at the expense of more complicated interfacing circuitry. As mentioned previously, the output signals supplied by the pyroelectric detectors are in pulsed form; each detector's pulse amplitude is proportional to the intensity of the incident beam sample, at the corresponding location on the main beam profile. Since this sample is provided by a rotating wand, the sample to each element arrives at a slightly different moment in time, as the wand sweeps across the face of the array. Therefore, a method was required of holding the output from each detector until all signals are valid. Fig. 4.4 illustrates the technique employed to accomplish this task. The signal from each element drives a circuit which detects and holds the peak pulse amplitude until it is reset. Resetting is controlled by a LED/photodiode trigger combination which detects the passing of the wand.

Sample waveforms from two individual detectors in the array, labeled A

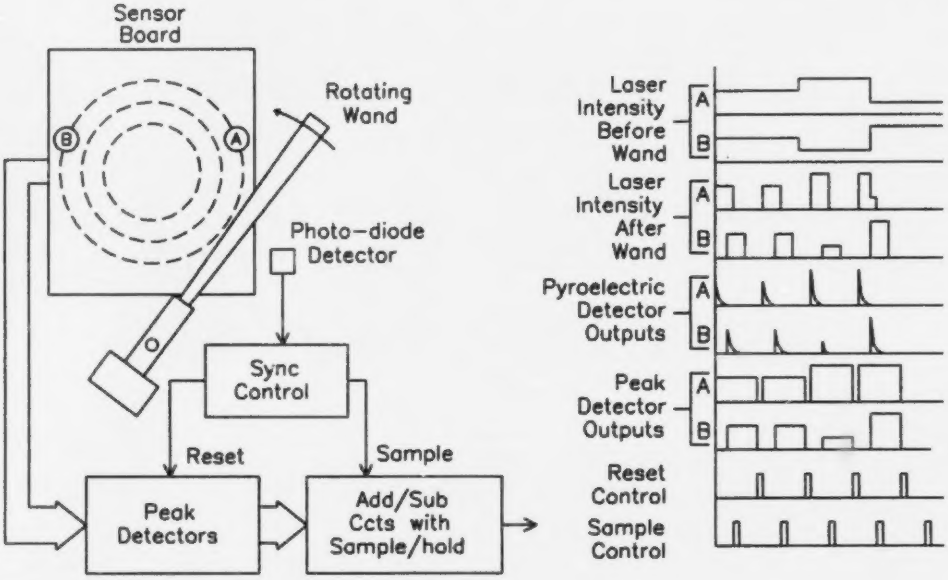


Fig. 4.4 Operation of Interface Circuits

and B, are shown in Fig. 4.4. As can be seen, the signal at element B is delayed with respect to that at A, in proportion to the wand's rotational speed. Each element's signal is held by its peak detection circuitry until the wand again passes the photodiode. Once each detector has received its portion of the beam profile, a control output from a synchronization circuit indicates that all outputs in the array are valid.

At this moment, sampling of the outputs may be performed. Upon completion of the sampling operation, the peak detection circuits are reset. Alternatively, analog adding/subtracting circuits may be employed prior to sampling, to obtain combinations of sensor outputs which are suitable for constructing the alignment error signals. This method was used to derive measures of the laser beam uniformity for the vertical and horizontal directions.

The peak detection circuit developed for a single sensor element is shown in Fig. 4.5.[34] Each pyroelectric assembly requires a resistor (R1) to set the load current in it's integral, common-source FET amplifier. Following the detector manufacturer's recommendations, this component was taken as 100 kΩ. Capacitor C1 removes the DC from the incoming signal, and R2 supplies bias current to op-amp U1. FET switch Q1 is employed to quickly discharge the holding capacitor (C2) after sampling is complete. General purpose LM-324 op-amps are used to minimize circuit board size; these units provide four operational amplifiers in a single 14 pin dip package.

To interface all array elements, 56 of these peak detector circuits were

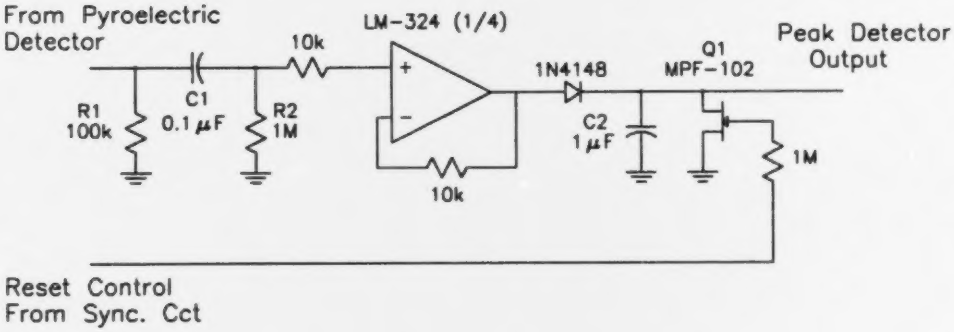


Fig. 4.5 Peak Detector Circuit

required. Breadboarding this large number of individual circuits was impractical; instead, a printed circuit board was designed to accommodate interface circuitry for two quadrants of the sensor array. Two of these circuit boards were then constructed to complete the interface between the sensor array and the control circuitry. In addition to the 56 peak detector circuits, summing circuitry was mounted on each board to calculate the quadrant sums from which the vertical and horizontal uniformity measures are derived.

4.5 Experimental Setup

The beam profile sensor described above was mounted on the PIE-3 laser, directly replacing the previous thermistor array. Circuit board size had been maintained identical to that for the thermistor array so that mounting modifications were unnecessary.

The interface circuits were inserted between the new sensor array, and the existing controller circuitry. A number of modifications[1] were also made to the controller circuitry to accomodate the sampled nature of the interface signals, and to take advantage of the faster response provided by the new feedback sensor.

The servos previously employed to provide alignment and input power level adjustment required no modifications other than speed adjustments when final tuning was performed.

At this time, a microcomputer based data-acquisition system was installed to improve the quality and efficiency of data collection. This system, shown in Fig. 4.6, was configured to provide monitoring of laser power and discharge current, as well as output signals from the new sensor array, and feedback control circuitry. Provision was also made to record the exact position of the motor-micrometer alignment servos, thus allowing the collection of accurate input/output data for the alignment control system.

As depicted in Fig. 4.6, a second rotating wand was also installed, to provide continuous on-line monitoring of the laser output power. The sample thus obtained was focussed into a commercial, thermopile type, laser power meter.[35] Observations revealed that power stabilization of the laser was best accomplished

---

[1] Schematic diagrams of these modifications are provided in appendix III. The controller configuration is discussed in greater detail later in this chapter.

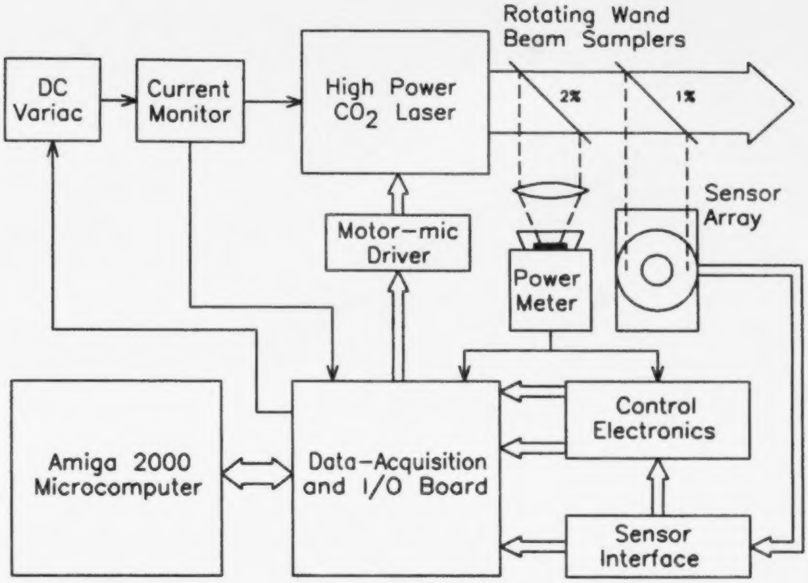


Fig. 4.6 Experimental Setup

using feedback from this power meter rather than using the sum of the sensor signals, as had previously been performed. This was attributed to the presence of unavoidable gaps between sensor array elements; it was thought that these gaps created slight errors in the sensor representation of total output power. It is likely that these errors were also present during the thermistor based experiments, but went undetected by the comparatively poor data collection methods employed at that stage.

In addition to data collection, this system was configured to allow computer control of the mirror alignment and power control servos. This provided a convenient means for collection of the data used to estimate the open-loop model parameters.

4.6 System Model

In general, before closed loop control of a system is attempted, it is desirable to obtain a model describing the plant. Once a suitable model has been constructed, design can then proceed, using either well known controller configurations or, if necessary, special controllers selected to suit the available servos and behavior of the open-loop system. In many situations a bang-bang controller structure is the most economical to implement.[36-38] When utilizing fixed speed servos, as in this application, a non-linear controller is also the most viable option. In any event, a first requirement is estimation of the parameters defining the open-loop model of the laser's alignment and power control subsystems.

In a fashion similar to that of previous experiments, a measure of the mode non-uniformity in the vertical and horizontal directions was defined by comparing the output intensity between opposing halves of the beam cross-sectional profile.[39,40] These comparisons were calculated from the sensor quadrant signal levels as follows:

$$Y_u = (A+B) - (C+D) \qquad (4.3)$$

$$X_u = (A+D) - (B+C) \qquad (4.4)$$

The model relating these uniformity measures to movement of the x and y alignment servos appears in Fig. 4.7. The model of Fig. 4.7 is divided into three sections, the horizontal and vertical axes of the resonator alignment system, and the power control portion of the overall laser system.

Consider first the horizontal axis of the alignment control section. Here the alignment servo is represented by an integrator with a single parameter, K_I (µm/s), which represents the motor-micrometer speed setting. This factor can be adjusted as desired; effectively allowing it to be used as a tuning parameter. Thus no estimation is required. The gain factor, K_{xx} (V/µm), describes the sensitivity of the horizontal uniformity measure to movement of the horizontal alignment servo. A second gain factor, K_{xy} (V/µm), accounts for the effect of servo movements along the vertical alignment axis. In practise it was found that this interaction parameter could be taken as zero; this was as expected since the resonator mirror mounts were configured to provide alignment about the X and Y axes. A similar situation for the vertical axis completes the model of the alignment portion of the system.

The DC current control variac is represented by three blocks. The first block, an integrator, models the motor used to rotate the variac shaft at a fixed speed in either the clockwise or counterclockwise direction. Following the integrator, a non-linear block is employed to model the backlash present in the gear

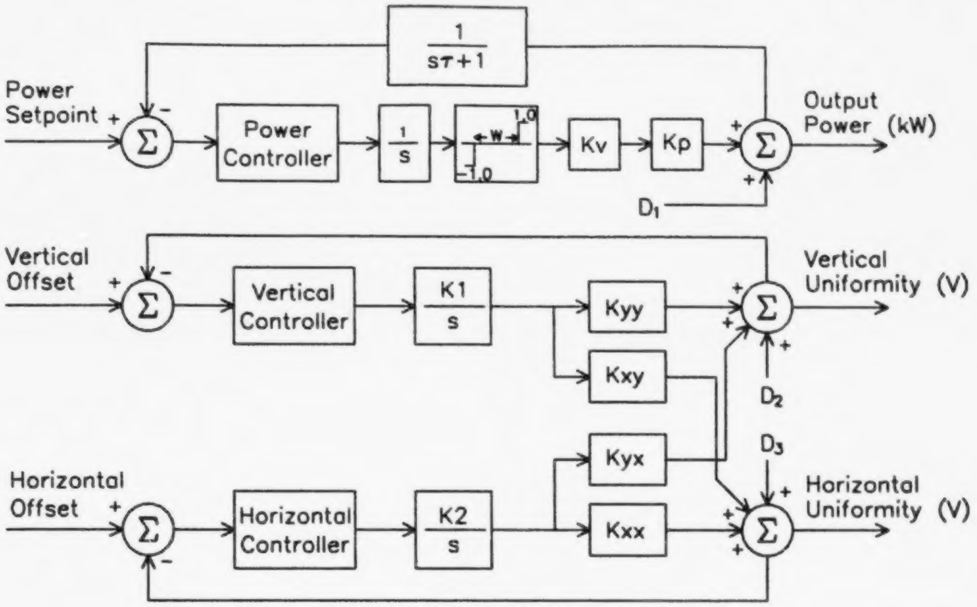


Fig. 4.7 Laser Alignment and Power Control System

coupling between the motor and variac shafts. Estimation of the dead-zone, W (seconds) is required for this block. The motor speed and the gear ratio of this block are effectively lumped together with the gain factor, K_v (Amps/s), which represents the rate at which the DC driving current to the laser changes when the variac is activated. An additional gain factor K_p (kW/Amp) is the conversion factor from pumping DC current to optical output power for the laser.

A first order block in the feedback path models the behavior of the laser power meter. Estimation of the time constant τ (seconds) of the thermopile based optical power measuring unit are required for this block. The gain of this block includes the 2% wand sampling ratio, and the power meter optical power to volts conversion ratio of 0.05 V/W, to provide an overall gain of 1 V/kW.

4.7 Parameter Estimation

4.7.1 Alignment System Parameters

Data was collected with the aid of the computer activated servos and data-acquisition system, to provide the parameter values defining the above model. The procedure used was as follows. After fixing mirror M2 at an appropriate alignment setting, primary mirror M1 was subjected to alignment changes about the horizontal and vertical axes. This was performed by first misaligning mirror M1 along the axis under test until power output from the laser diminished to half its initial level. The direction of movement was then reversed and data collection resumed until the mirror was again misaligned to the half power point on the opposite side of the center position. Input current to the laser was held fixed at a convenient level during these alignment changes, and the micrometer position together with data from the beam profile sensor, was recorded at each sample point.

As is usually the case in physical processes, a large amount of noise contamination was present in the data. Therefore, before attempting to use this data for parameter estimation, a digital filtering routine was employed to remove the high frequency noise. This filter was obtained by performing a bilinear transformation[2] on a 3rd order analog filter. The Matlab program[3] was then employed to perform off-line low-pass filtering of the data with a cutoff frequency of 0.25 Hz. The raw data, along with the resulting filtered data, is plotted in Fig. 4.8.

It is immediately apparent that the uniformity measures exhibit a linear response over a relatively wide range about the center aligned position. A simple least squares fit was performed within the linear region of each of these characteristics to provide estimates of the K_{xx} and K_{yy} parameters in the model of Fig. 4.7. A similar procedure was performed to determine the interaction terms K_{xy} and K_{yx}; the results were not particularly interesting, and so have been omitted from

---

[2] Details of the design procedure used to obtain this filter are provided in appendix I.

[3] One particularly useful MATLAB function effectively doubles a filter's order by filtering the data first in the forward direction, and then reversing the sequence and repeating the process. This approach provides a filtered output which exhibits precisely zero phase distortion. Startup transients are eliminated by matching initial conditions.

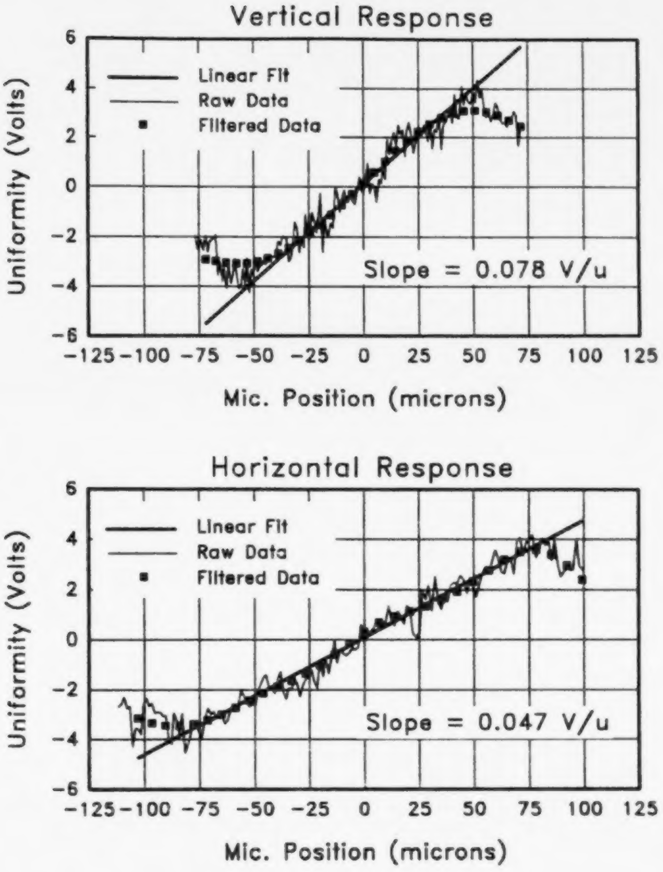


Fig. 4.8 Estimation of the Alignment System Parameters

the plots to avoid clutter. In both cases, the interaction terms were small enough to allow their elimination from the model.

4.7.2 Power Control System Parameters

Similar input/output data were collected to estimate the parameters defining the power control portion of the system. With the laser in an aligned condition, data to estimate these parameters were collected by activating the DC variac to induce an input current level change of approximately ± 2.0 amps about the chosen operating point of 20.0 amps. This variation in current was sufficient to create a corresponding change in output power of about ± 500 W. Measurements from the DC current monitor and the laser power meter together with the status of the variac control signal, were recorded at each sample point.

Estimation of the non-linear block was performed with the aid of a commercially available simulation package (TUTSIM) which provides for simulation of both linear and non-linear elements.[41] Parameter estimation is performed by first constructing a model of the elements to be estimated as shown in Fig. 4.9. By comparing the model output to the measured data, a measure of the model/process mismatch can then be defined. In the present situation this error measure was calculated as the integral of the squared difference between the model output and the raw data.

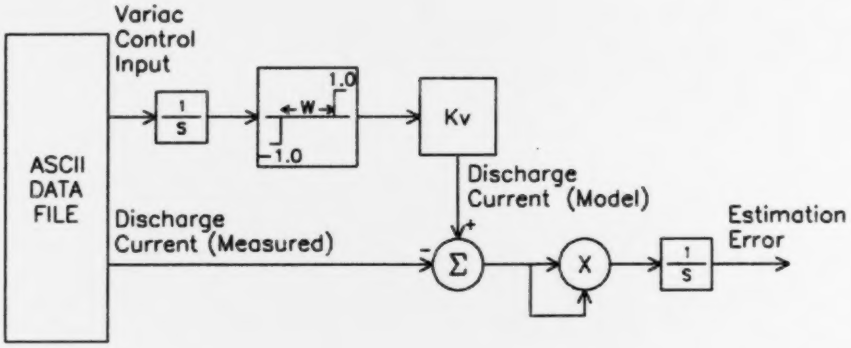


Fig. 4.9 Variac Estimation Procedure

After the model has been entered in appropriate form, the TUTSIM program iteratively modifies the desired parameters until the final value of the error measure is less than a user-defined amount. The program was employed to estimate the dead-zone width W (seconds) and the K_v (Amps/s) gain parameter.

The remaining parameters, K_p and τ, were estimated using the batch least squares method.[42] A discrete first order model was proposed as follows:

$$P_o(n) = a_1 P_o(n-1) + b_1 I_d(n-1) \qquad (4.5)$$

where: P_o = Laser output power
I_d = Input driving current
a_1, b_1 = parameters to estimate
n = sample number

A least squares estimate of the parameters a_1 and b_1 is given by:[4]

$$\Theta = (\Phi^T \Phi)^{-1} \Phi^T \mathbf{P}_o \qquad (4.6)$$

where:

$$\Theta = \begin{vmatrix} a_1 \\ b_1 \end{vmatrix} \qquad \mathbf{P}_o = \begin{vmatrix} P_o(2) \\ P_o(3) \\ P_o(4) \\ \cdot \\ \cdot \\ \cdot \\ P_o(N) \end{vmatrix} \qquad \Phi = \begin{vmatrix} P_o(1) & I_d(1) \\ P_o(2) & I_d(2) \\ P_o(3) & I_d(3) \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ P_o(N-1) & I_d(N-1) \end{vmatrix}$$

N = the number of samples in the I/O vector

It is easily shown that the equivalent continuous model for the power meter is described by:

$$\tau = \frac{-T_s}{\ln(a_1)} \quad \text{and} \quad K_p = \frac{b_1}{1 - e^{-(T_s/\tau)}} \qquad (4.7, 4.8)$$

where: T_s = sampling period (seconds)

---

[4] Since the batch least squares method of parameter estimation is well described in the literature, only the final result is presented here. Equation 4.6 yields parameter values which define the model that fits the I/O data vector in a least squares sense.

Fig. 4.10 shows that the model response follows the measured data closely for both the variac and the power meter. A block diagram of the completed model appears in Fig. 4.11. The alignment portion of the model is simplified considerably by the elimination of the two interaction parameters k_{xy} and k_{yx}.

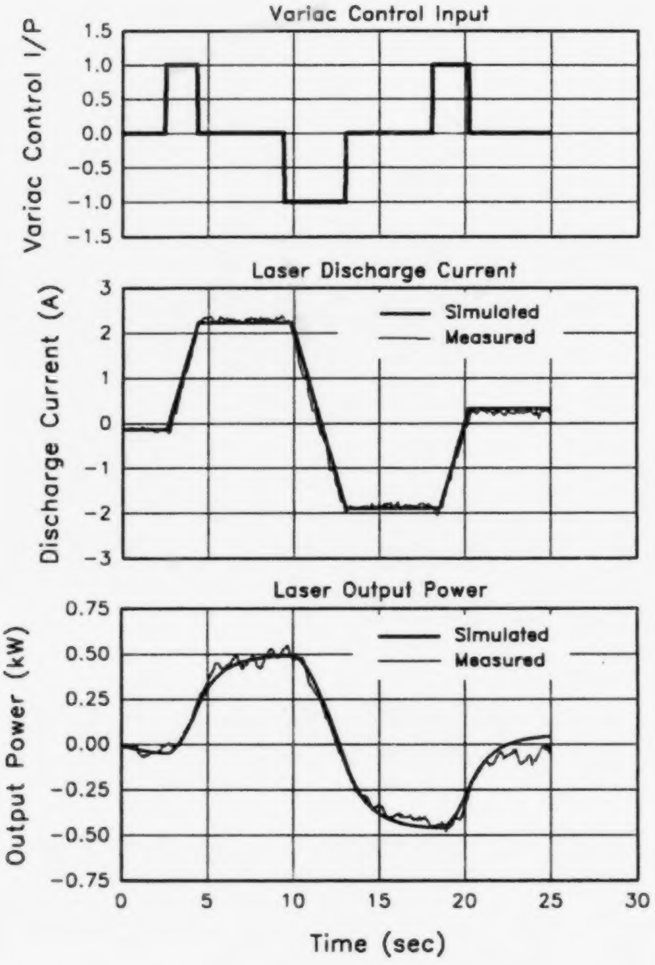


Fig. 4.10 Power Control System Estimation Results

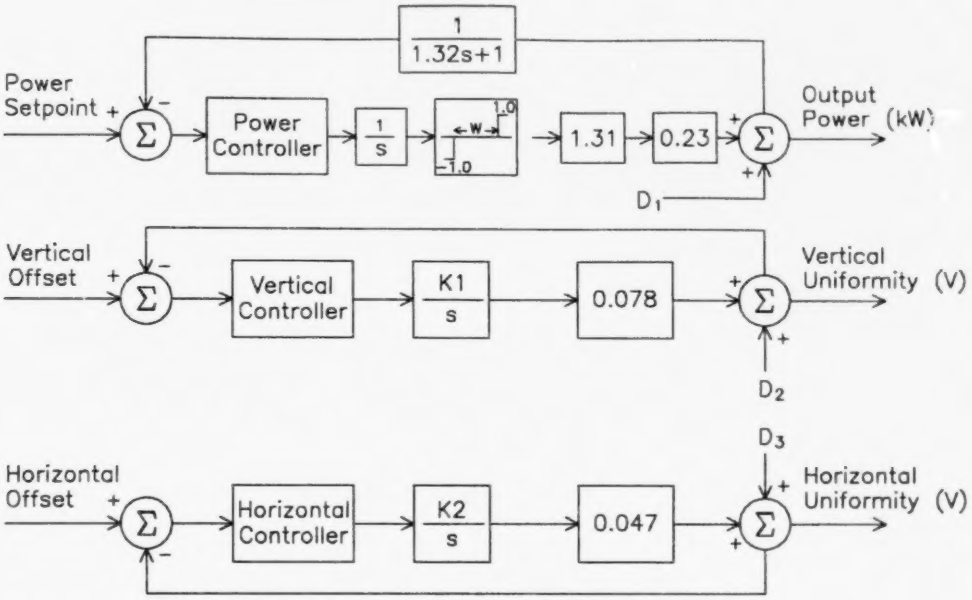


Fig. 4.11 Laser Alignment and Power Control System Model with Estimated Parameters

4.8 Controller Design

Before attempting closed loop control of the high power laser, computer simulations were programmed using the TUTSIM package.[5] This simulation was used for evaluation of different controller structures as well as for obtaining initial tuning parameters. The absence of interaction terms between the two alignment axes allowed independent simulation of each alignment control loop, using the same program. Examination of each axis was then performed by simply modifying the appropriate parameters to accurately reflect the axis under consideration. A slightly modified version of the program was also employed to perform a simulation of the power control loop.

The presence of fixed speed servos in each loop of the system restricts the controller structure to a bang-bang type. This restriction is not in itself objectionable, since controllers with dead-zones can be used to eliminate undesirable servo movements induced by system noise.[43-45] With the aid of the simulation program, various controllers were experimented with until the controller structure of Fig. 4.12 was finally selected.[6]

---

[5] TUTSIM (Twente University of Technology SIMulation Program) provides for simulation of both linear and nonlinear elements. It is programmed in similar fashion to an analog computer, but provides added versatility and accuracy. Additional detail regarding these simulations is available in appendix II.

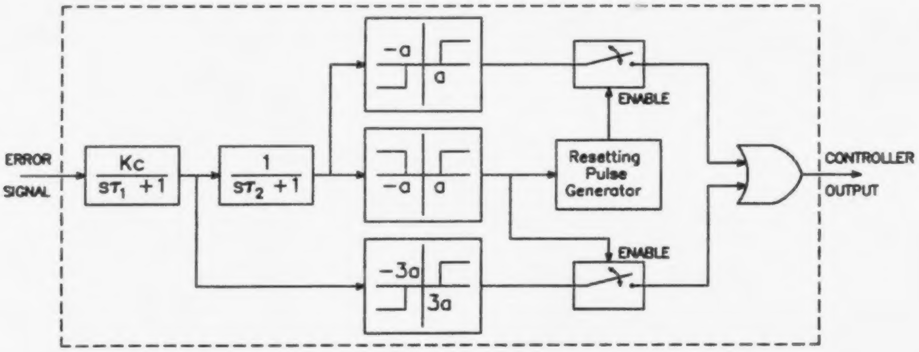


Fig. 4.12 Non-linear Controller Structure

Two speed control is implemented by selectively controlling the duty cycle of the control servos, based on the magnitude of error signal present at the controller input. As shown in Fig. 4.12, two levels of filtering were employed. The first level was designed with a faster response so that control movements at high servo speeds could be applied when large errors are present. Such a situation occurs at startup, or when large changes in power setpoint are made. The second level of filtering was designed with a slower response, to provide a smoother signal. Only intermittent regulatory servo movements are based on the output from this filter.

The approach described above allows the use of higher system gains. In this manner, good regulation at steady state is provided, while still retaining fast transient response during startup periods. A wider dead-zone was utilized on the "fast" portion of the controller to compensate for the higher level of noise present in the output from the fast filter. Additional noise immunity was obtained by disabling the fast controller until the slower error signal drifted outside its control window.

---

[6] The controller structure of Fig. 4.12 also proved to be effective in the power control loop, in spite of the presence of the backlash element.

4.9 Experimental Results

4.9.1 Alignment Controller Tuning

Filter time constants were selected using the data collected for parameter estimation purposes. Using a digital filter routine, various filter cutoff frequencies were evaluated to achieve an acceptable compromise between error signal noise level, and phase lag. The resulting filters provided cutoff frequencies of 0.78 and 0.16 Hz, respectively. The controller gains were experimentally adjusted as high as possible while still allowing the slow error signal to remain inside the control window under steady state conditions.

The duty cycle of the slow controller was fixed at 5% and the micrometer speeds were used as the final tuning parameter. Tuning was initially performed using the TUTSIM computer simulation. Priority was given to minimizing servo movements rather than to achieving exceptionally fast response. The resulting response resembled a critically damped system.[46] With a gain of 4.0 in each controller, suitable micrometer speeds were found to be 10 and 15 μm/s, for the vertical and horizontal controllers, respectively.

4.9.2 Power Controller Tuning

In the case of the power controller, the first stage of filtering was actually provided by the thermal time constant (0.12 Hz) of the power meter. The second filter cutoff frequency was selected as 0.16 Hz, identical to that of the alignment controllers. A simple experiment was next performed to determine the minimum resolvable current step size that the variac could accommodate. Based on this result, the pulse width of the slow controller was set to 150 ms, and the gain and duty cycle were adjusted to obtain control without limit cycle behavior, under steady state conditions. Finally, the fast portion of the controller was tuned by adjusting the dead-zone width to eliminate overshoot. In the case of the power controller, a large amount of overshoot would be dangerous to the laser components, particularly the expensive zinc-selenide output window. Again, the TUTSIM simulation was initially used to obtain suitable tuning parameters. The best performance was achieved using a gain of 25.0, a slow controller duty cycle of 7.5% and a fast controller dead-zone of ± 6.0.

4.9.3 Transient Response

Observations of the alignment controllers performance revealed that the controller could easily provide proper alignment upon startup of the laser, provided that at least a small portion of the beam was present. A maximum of about 15 s was required to attain a fully aligned condition from virtually any extreme of misalignment. Step response evaluation of the alignment controller was obtained in the following manner. First the controller was disabled and then an intentional misalignment was manually introduced in both the vertical and horizontal directions. Here, mirror M1 was moved slightly beyond the linear region of the uniformity measures. Finally, automatic control was re-engaged. For the power controller, the response to an induced step change from 1.0 to 2.0 kW was examined. Both responses are plotted in Fig. 4.13, together with the simulated output for comparison. In the case of the alignment response, the output signal from the slow filter was plotted. For the power system, the output from the power meter was plotted.

4.9.4 Steady-State Response

Steady-state performance of the laser was monitored during two 15 minute runs: first in open-loop mode, and then with the automatic alignment and power controller engaged. During the first run the controller was allowed to stabilize the laser for the first minute of operation. The controller was then disengaged and the laser's performance monitored until the 10 minute mark in the run. At 10 minutes the controller was returned to automatic and allowed to again stabilize the laser for the remainder of the run. For the second run, the controller was engaged upon startup and allowed to stabilize the laser for the entire duration. Output power level and uniformity signals for the vertical and horizontal directions are plotted for each run in Figs. 4.14 and 4.15, respectively.

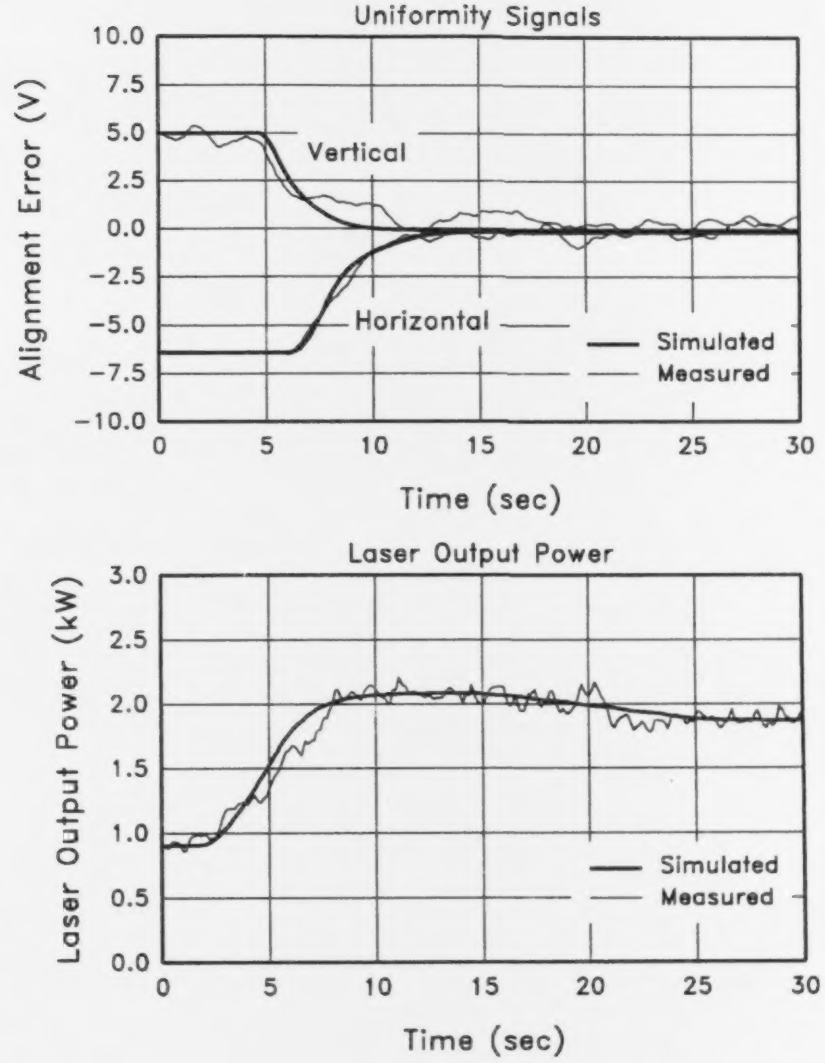


Fig. 4.13 Transient Response of Closed-Loop System
(Simulated and measured)

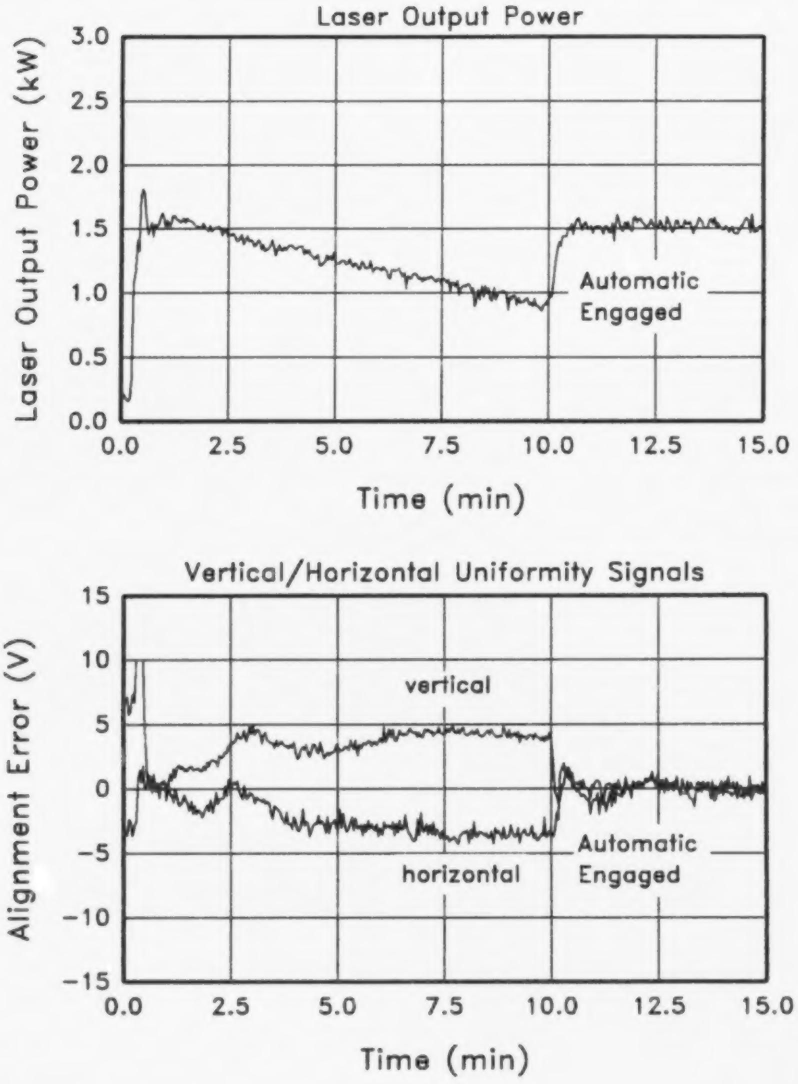


Fig. 4.14 Long-Term Open-Loop Performance

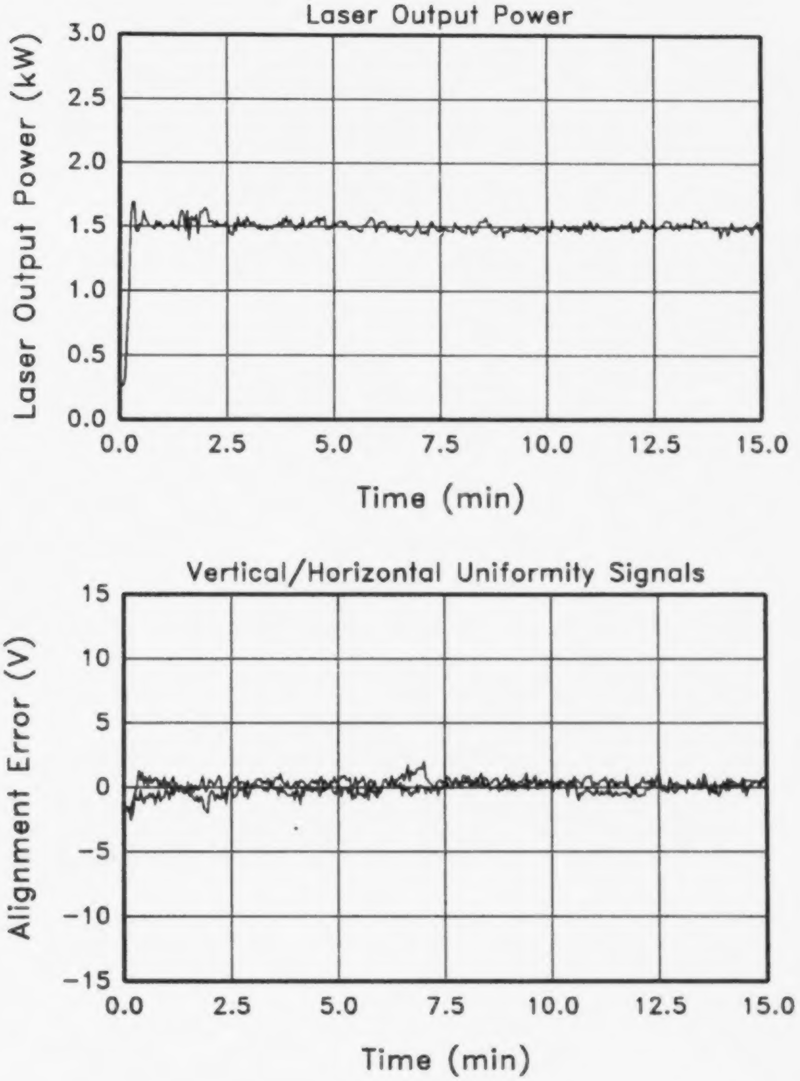


Fig. 4.15 Long-Term Closed-Loop Performance

4.10 Discussion

4.10.1 Performance

The two graphs of Fig. 4.13 reveal a greatly improved performance, as compared to that of the original thermistor based system. Specifically, transient duration was reduced from a 2 minute response, characteristic of the thermistor sensor, to less than 10 s with the new pyroelectric system.

Fig. 4.14 clearly reveals that without closed loop control the output power of the laser begins to decay almost immediately after the controller is disengaged and continues this trend until the controller is engaged whereupon the original power level is quickly restored. In similar fashion, the two uniformity measures also undergo a rapid drift from the desired center position, until the controller is allowed to reinstate proper resonator alignment.

The plot of Fig. 4.15 illustrates the superior laser performance achieved when both beam uniformity and power level stabilization systems are engaged continuously. Visual observation of the laser's output mode quality provided additional evidence of the superior performance obtained by the closed-loop system. Close examination of the data reveals that the control system stabilizes the output power level to within approximately ± 100 W of the desired value. Tighter power regulation was not possible with the present laser; being limited by the smallest step size attainable using the variac-based DC power supply.

4.10.2 Stability

A major consideration in the development of any control system is stability. Many physical systems can be damaged by oscillations, especially those which grow without bound. Although, the stability of non-linear systems has been a subject of extensive study, a universal method of determining the stability of these systems has not been found.[47] In many systems using relay type controllers, it is quite possible to obtain a system which exhibits stable oscillations. One example of this is found in the simple on-off action of a common temperature control system. In the alignment and power control system developed here, such oscillations would degrade system performance, and would lead to unnecessary wear of the alignment and variac servo-mechanisms. The main goal of a stability analysis here is to determine if limit cycle oscillations may occur in either of the alignment or power control loops.

A number of analytical methods are available for determining the existence of limit cycle oscillations in relay control systems, like the one used here.[47-49] It would be quite acceptable to apply one or more of these methods. However, since a direct simulation of the system was already available, it was convenient to utilize this program to examine the effect of controller gains on system stability. Using the simulation, it was seen that alignment controller gains as high as 30 were possible before the onset of a limit cycle oscillation. Gains higher than 30 progressively increased the amplitude of the stable oscillation. As reported earlier, a gain of 4.0 was chosen experimentally; this relatively low gain ensured stability.

Similar behavior was observed in the power control loop. The simulation exhibited a limit cycle oscillation with power controller gains of 41 or higher. Consequently, the experimentally selected gain of 25 for this control loop provided an adequate stability margin. In both the alignment and power control loops, experimental observations indicated that the system was unconditionally stable.

4.11 Summary

As a result of the poor transient performance provided by the thermistor based system, the development of an improved sensor, based on individual pyroelectric detectors, was undertaken. These small pyroelectric units were more durable than the large, single crystal detector, used in the low power experiments. The new sensor array was also considerably less expensive, requiring a total outlay of less than one tenth of the cost of the large quadrant detector.

The pulsed nature of the signals provided by the new sensor necessitated the development of a more complicated interface to drive the control circuitry. A printed circuit board was designed to provide the necessary peak detection and summing circuitry. To fulfill the interface requirements, two of these circuit boards were constructed.

At this point, a microcomputer based data-acquisition system was also installed to provide a more accurate method of data-collection. This system enabled computer controlled collection of input/output data which was subsequently used to construct a mathematical model describing the power and alignment systems.

After a suitable model of the system was developed, a computer simulation was programmed to facilitate the design of a controller for the system. A primary factor complicating controller design was the poor performance provided by the alignment servos. This was caused by the relatively heavy load conditions present in the PIE-3 optical system. This difficulty was overcome by operating the motor-micrometers at a relatively high, fixed speed, and then using pulses to obtain small alignment changes at slow rates. This approach required the design of a non-linear controller; the computer simulation provided a convenient means for testing the performance of various controller structures. The resulting two-speed controller provided good transient response with minimum servo movement under regulatory conditions. This same controller also proved quite adequate for use in the power control loop.

Finally, data was obtained documenting the transient and steady-state performance of the new closed-loop system. This data clearly revealed the superior response achieved using the new sensor. Specifically, settling time was reduced from a 2 minute response, characteristic of the thermistor sensor, to less than 10 s with the new pyroelectric system. It can be concluded that this system, even at its present level of sophistication, would dramatically improve the performance of high power lasers currently operating without closed-loop alignment and power stabilization.

However, the problems affecting high power laser resonators are not confined to a single optical element. It is recognized that all of the resonator components are affected by thermal distortion. Although it would be quite difficult to compensate every element individually, it was thought that an additional improvement in performance could be obtained, by extending the system to control the second primary mirror. A discussion of the methodology employed to accomplish this task is provided in Chapter 5.

CHAPTER 5

FEEDBACK CONTROL OF BEAM POINTING ANGLE

Although a considerable improvement in laser performance has been realized with the single mirror control system discussed in Chapter 4, there is strong motivation to extend the experiments to include control of the second primary mirror. Inasmuch as thermal distortion affects both resonator elements, it is reasonable to expect that further improvement in performance can be obtained by compensating for the distortion of each mirror separately. Furthermore, as power input to the discharge is increased, the refractive index in the medium begins to exhibit an increasingly significant gradient, across the mode volume. The only means of compensating for this effect is to actively align the resonator mirrors.

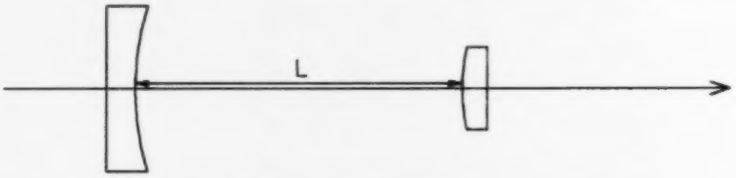
Unfortunately, when mode uniformity begins to deteriorate, it is difficult to attribute the degradation to misalignment of a single resonator element. Experiments thus far have demonstrated that a reasonably uniform mode can be maintained by aligning the convex mirror. However, by using only one element to compensate for thermal distortion of both primary mirrors, the effect upon beam pointing angle is disregarded. Consequently, further study was undertaken to determine if it was possible to perform on-line detection of the beam pointing angle, using the existing pyroelectric sensor.

To accomplish this task, 2 new error signals were defined. These signals were designed to indicate vertical and horizontal deviations, from center, in the position of the beam sample. The system model was then extended to describe the behavior of these position measures, in response to alignment changes of the second primary mirror. A relay controller was subsequently designed to perform continuous, beam pointing angle stabilization on the 5 kW PIE-3 laser, based on these new error signals.

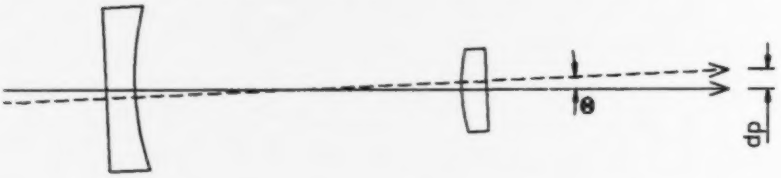
The following section describes the technique utilized to monitor the beam pointing angle. Estimation of the additional model parameters is then discussed, followed by a description of the beam position feedback controller. Finally, experimental results documenting the performance of the completed two mirror control system are presented.

5.1 Beam Steering in Optical Resonators

The concept of beam steering in laser resonators is well described in the literature.[50-52] As illustrated in Fig. 5.1, the optical axis is shifted when both mirrors are misaligned. Mode uniformity will remain essentially undisturbed, provided the optic axis and a region of diameter $\sqrt{\lambda L}$ lie within the boundaries of both mirrors. As shown in Fig. 5.1, if the center position of the beam is monitored at a fixed distance from the output of the resonator, the pointing angle is effectively determined. In the case of annular output resonators, which are commonly employed in high power industrial lasers, it is necessary to devise a method of detecting the position of the annulus.



a) Optic Axis in a perfectly aligned resonator



b) Displaced optical axis in a parallel but misaligned resonator

Fig. 5.1 Beam Steering Effect in a Laser Resonator

Fortunately, the 5 kW PIE laser used in the experiments was equipped with a thermoluminescent viewing screen.[1] This screen provided the opportunity to visually monitor the effect of resonator alignment changes upon beam position.[53] This procedure revealed that beam position could be significantly altered by manually aligning the concave mirror, while simultaneously maintaining beam uniformity with the convex mirror. These observations suggested that, provided a suitable method of monitoring beam position could be found, alignment of both the concave and convex mirrors could be performed automatically.

---

[1] This device consists of a latex rubber sheet coated with a phosphor which fluoresces with a bright yellow color when illuminated with UV light. Simultaneous heating by an infrared source, such as a laser beam, quenches this fluorescent effect, causing the screen to darken in direct proportion to the amount of infrared energy present.

5.2 Sensor Interface Circuit Modification

As described previously, the interface circuits provided 4 signals proportional to the optical intensity in each quadrant of the laser's annular output beam. Two error signals describing the vertical and horizontal uniformity of the laser mode were subsequently extracted from these quadrant sums. These two signals were then employed to perform feedback control of mode uniformity.

To detect changes in beam position, additional summing circuits were utilized to provide 12 signals, each representing the beam's intensity in 1 of 3 concentric arcs within each quadrant, as shown in Fig. 5.2. From these signals, measures of the change in vertical and horizontal position of the beam sample were defined as follows:

$$Y_p = (A_2+B_2+A_3+B_3+C_1+D_1) - (A_1+B_1+C_2+D_2+C_3+D_3) \qquad (5.1)$$

$$X_p = (A_2+D_2+A_3+D_3+B_1+C_1) - (A_1+D_1+B_2+C_2+B_3+C_3) \qquad (5.2)$$

Fig. 5.2 illustrates the operating principle behind these position measures. In Fig. 5.2 (A), a uniform annulus is present on the sensor array. The intensities in the corresponding arc sections of each quadrant are equal under this condition and so, Y_p and X_p are held at zero. Now consider that the position of the annulus shifts slightly upward, as in Fig, 5.2 (B). The signals A_2, B_2, A_3, B_3, C_1, and D_1 increase while A_1, B_1, C_2, D_2, C_3, and D_3 decrease. The Y_p measure becomes positive, indicating an upward shift in the beam position. A similar analysis applies for beam motion in the horizontal direction. One restriction that becomes immediately apparent is the need for the annulus to remain relatively uniform. This condition can be somewhat satisfied by performing concave mirror alignment at a reduced rate. Using this procedure helps to ensure that the convex mirror controller is able to continuously maintain a uniform output mode.

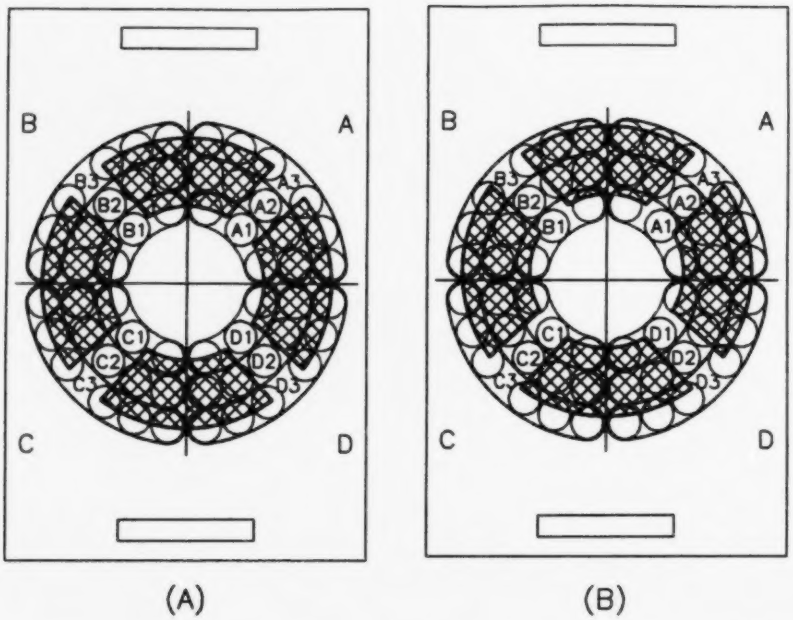


Fig. 5.2 Sensor Grouping for Position Detection

5.3 Extended Plant Model

The mathematical model relating the new position error measures to horizontal and vertical movements of the concave mirror alignment servos, is shown in Fig. 5.3. As shown in the diagram, interaction between the uniformity and position measures was assumed to be negligible. Instead, the uniformity measure is related entirely to the convex mirror, and the position measure to the concave mirror. This approach is validated, by requiring alignment of the concave mirror to be performed at a reasonably slow rate. As mentioned earlier, this allows the uniformity controller to continuously compensate disturbances to uniformity caused by changes in the alignment of the concave mirror. Any residual interaction between the two error measures was treated as disturbances. Disturbances to power level, uniformity and beam position are shown as the "D" inputs at the output summer.

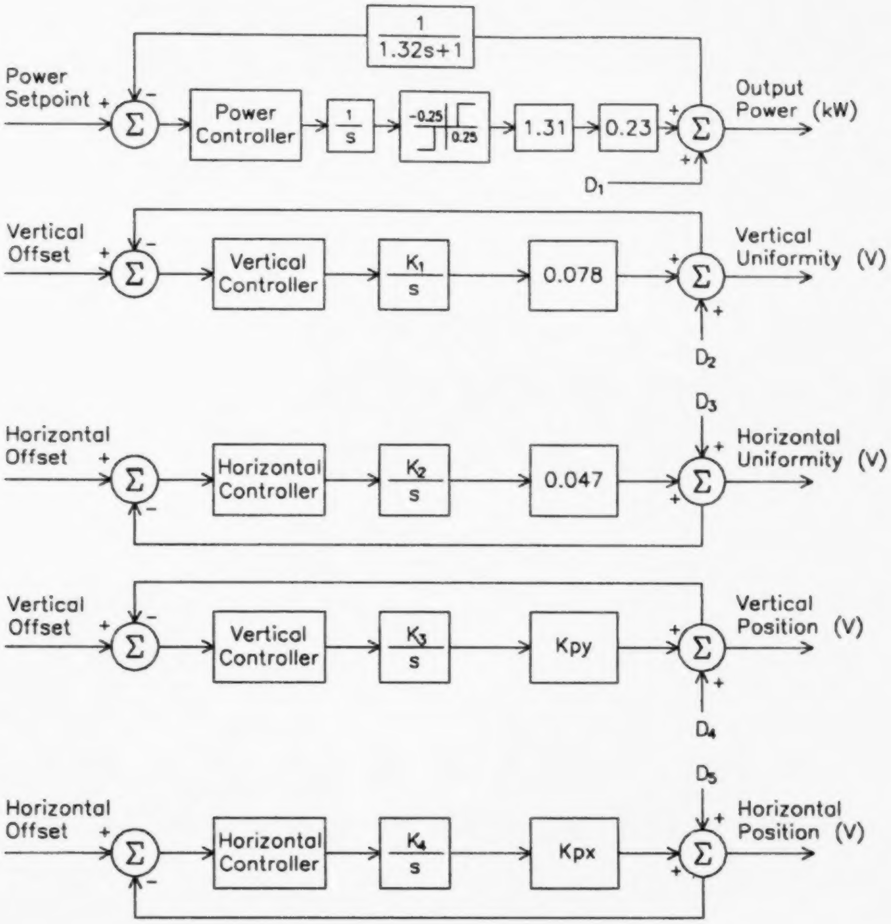


Fig. 5.3 System Model for Two Mirror Control

In similar fashion to the uniformity portion of the model, the alignment servo for each axis is represented by an integrator with a gain factor, to describe the speed setting of the motor-micrometer. As before, these speed factors are user adjustable, allowing them to be utilized as tuning parameters. The relationships between the vertical and horizontal position measures, and the positions of the concave mirror alignment servos, is represented by the constants K_{py} and K_{px} (V/μm), respectively. These two gain factors are the new parameters which require estimation.

5.3.1 Parameter Estimation

Data to estimate the K_{py} and K_{px} parameters were collected using the following procedure. First, the motor-micrometers for the concave mirror were adjusted to a fixed speed of 10.0 μm/s. This speed was sufficiently slow to permit the convex mirror control system to continuously maintain mode uniformity as the concave mirror was aligned. With the uniformity controller on automatic, the concave mirror was then manually aligned, so that both position measures equaled zero. From this state, the vertical axis micrometer on the concave mirror was jogged about the center position by ± 150 μm; this distance was sufficient to observe a significant variation in the position measure signals. The reading from the motor-micrometer, and the corresponding position error signal, was recorded at each data point. An identical procedure was used to collect data for the horizontal direction.

A first order least squares fit, with the micrometer reading and the corresponding position measure as independent and dependent variables, respectively, provided the gain parameters. The K_{py} value was estimated as 0.019 V/μm, and K_{px} was estimated as -0.015 V/μm. Fig. 5.4 compares the measured and model responses for the vertical and horizontal directions. The completed control system model appears in Fig. 5.5. In comparison to the uniformity gain factors, the position measure gains are much less sensitive to changes in resonator alignment. This was as expected since visual observation revealed that a relatively large change in concave mirror angular alignment was required to produce a noticeable shift in beam position.

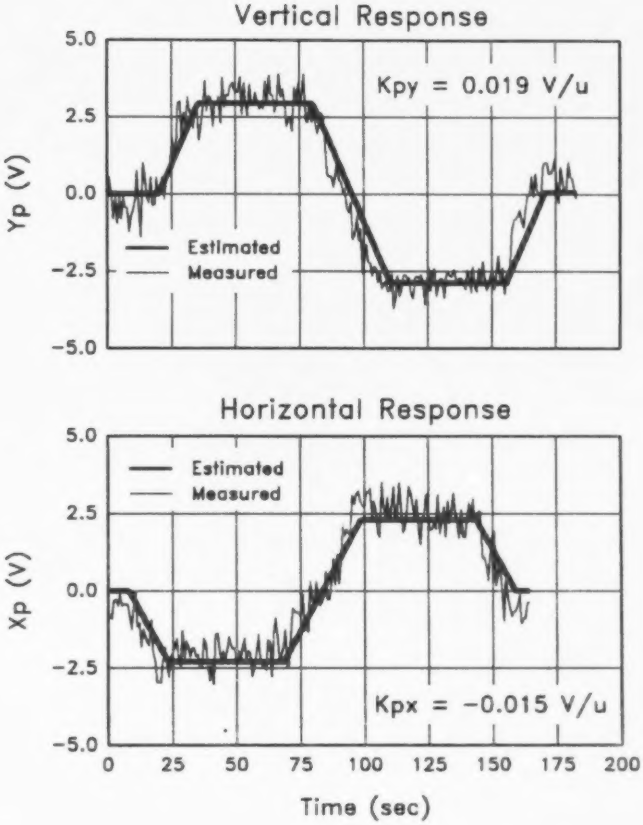


Fig. 5.4 Estimation of Position Measure Gain Factors

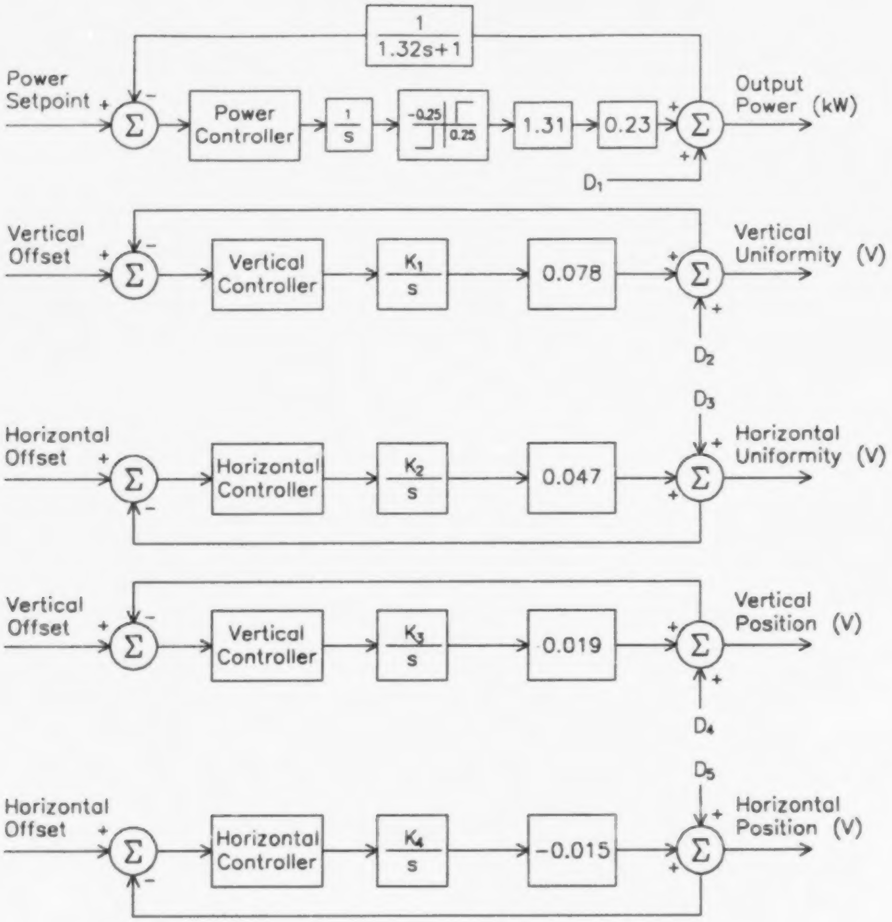


Fig. 5.5 Completed PIE-3 Laser System Model

5.4 Controller Design

Experimentation with alignment of the concave mirror revealed that a human operator could easily accomplish manual control of the beam position by monitoring the new error signals. This was performed by setting the uniformity controller to automatic, and then aligning the concave mirror so as to maintain the vertical and horizontal position measures as close to zero as possible. These experiments suggested that the simple relay controller of Fig. 5.6 would suffice for

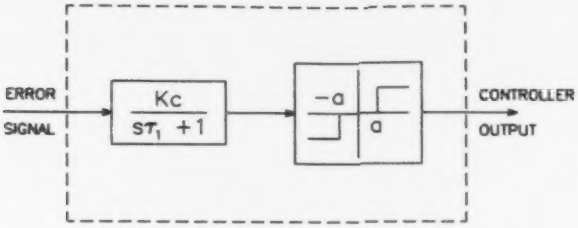


Fig. 5.6 Beam Position Relay Controller

stabilizing the beam position. The low-pass filter preceding the relay was designed with a cutoff frequency lower than that employed in the uniformity controller. This helped to ensure that the position error signals would be relatively undisturbed, by changes in mode structure which occur as the convex mirror controller works to stabilize the annulus uniformity. After some experimentation, a cutoff frequency of 0.05 Hz was selected.

5.5 Experimental Results

5.5.1 Controller Tuning

The dead-zone in the position controller described above was set to ± 0.6 V and the gains were adjusted as high as possible, based on the level of noise present in the filtered error signal. It was found that a gain of 10 provided good sensitivity to movements of the concave mirror while still remaining inside the dead-zone in steady-state. Micrometer speeds of 8 and 13 μm/s were then selected for the vertical and horizontal directions, respectively. These micrometer speeds were slow enough that the uniformity controller was able to maintain the beam fairly close to uniform during alignment adjustments to the concave mirror.

5.5.2 Transient Response

Step response of the position controller was obtained using a similar procedure to that employed to evaluate the uniformity controller. Uniformity control was continuously enabled during the test procedure. The position controller was disabled, and the concave mirror was initially misaligned along both axes by

150 μm.[2] The position controller was then returned to automatic and the data of Fig. 5.7 was collected as the controller restored proper alignment. Results were obtained for both the vertical and horizontal axes.

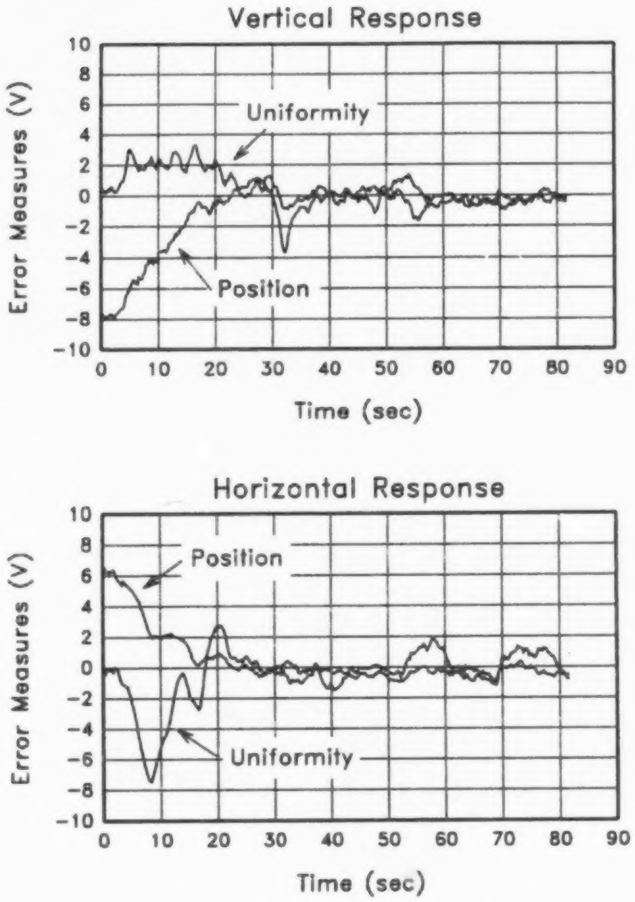


Fig. 5.7 Transient Response of the Position Control System

---

[2] By considering the moment arm of the mirror mount assembly, and the ratio of the differential micrometer, it can be shown that one micron of micrometer movement corresponds to a change in mirror angle of 0.282 μrad. Consequently, a micrometer displacement of 150 μm represents an angular misalignment of 42.3 μrad.

5.5.3 Steady-State Response

The steady-state performance of the combined uniformity, position and power control systems was obtained in 2 separate runs. In each run, performance of the laser was monitored from a cold start condition, for a period of 15 minutes. The first run was performed with only the uniformity and power controllers engaged. For the second run, the beam position controller was also engaged for the

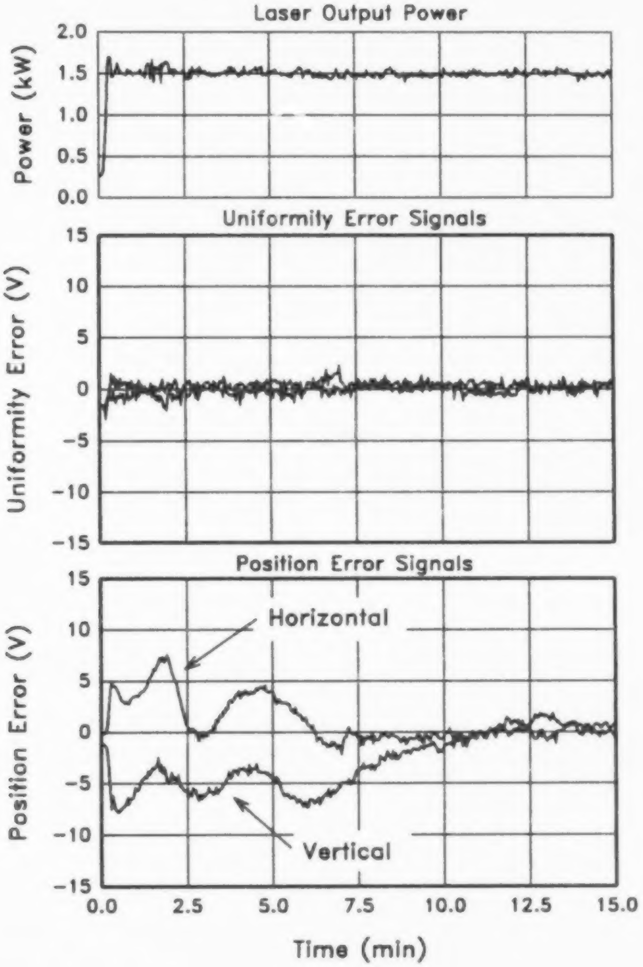


Fig. 5.8 PIE-3 Steady-State Performance with Power and Uniformity Control but without Position Control

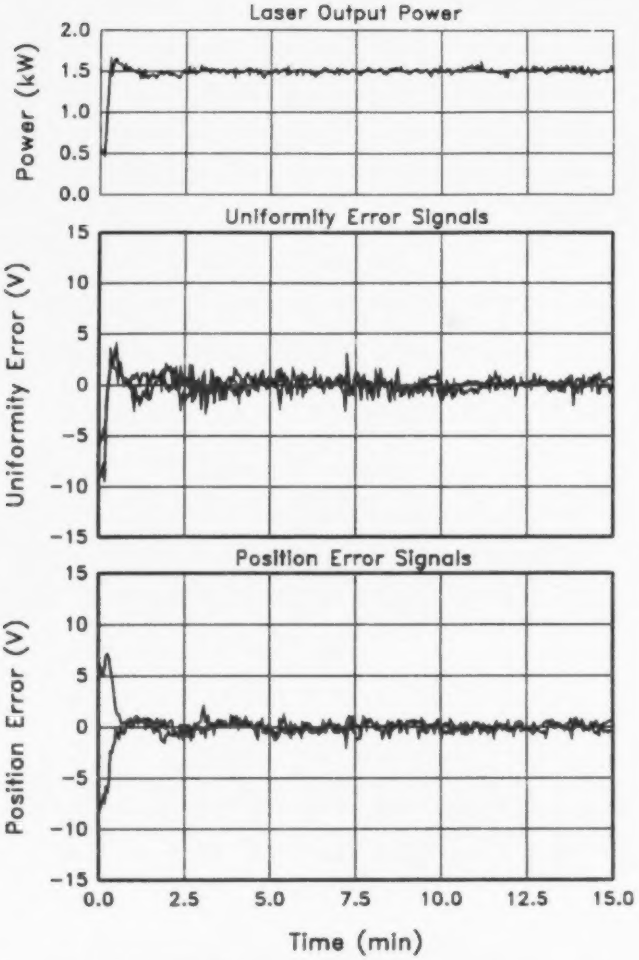


Fig. 5.9 PIE-3 Steady-State Performance with Power, Uniformity and Position Control

duration of the run. The uniformity and position error signals are plotted together with output power in Figs. 5.8 and 5.9.

5.6 Discussion

5.6.1 Position Controller Performance

The plots of Fig. 5.7 demonstrate that the position controller is able to return the beam pointing angle to the center position, from a rather extreme misalignment, within approximately 20 seconds. The graphs also reveal the disturbances to beam uniformity which occur as the concave mirror undergoes alignment changes. Note that these disturbances are more pronounced along the horizontal axis. This result was deliberately induced in this particular test, by choosing the speed of the position control servo slightly higher than the speed of the uniformity control servo. Although the system eventually settles, it was observed that these disturbances could be reduced significantly by choosing slower micrometer speeds for position control than for uniformity control.

Long term stability of the beam pointing angle is significantly improved, as evidenced by the plots of Fig. 5.8 and 5.9. As can be seen in Fig. 5.8, beam uniformity and output power level are well stabilized without position control. However, during the first 10 minutes of the run, the beam position wanders, as the temperature of the laser vessel and optical system progresses towards its steady state operating level. It is of interest to note in Fig. 5.8, that at approximately the 10 minute point in the run, the beam position error returns to near zero. This was a direct result of a previous run, in which the position controller was engaged continuously. Consequently, when the second run was performed without position control, the concave mirror was almost perfectly aligned for the warm condition.

Fig. 5.9 demonstrates the improved beam position stability which is achieved when the automatic control system is engaged upon startup. The most dramatic improvement in laser performance is clearly provided during the initial warm-up period. Beam uniformity, position, and power level are held very close to the desired setpoints for the entire run. It is also evident that when operated continuously, the effect of the position controllers upon beam uniformity is minimal.

After observing the performance of the controller for a number of runs, it became apparent that the controller could be depended upon to maintain a consistent output beam, not only during a single run, but between successive runs. It was seen that even if both primary mirrors were severely misaligned, the automatic controller could restore correct beam uniformity and position. Of course, at least a small portion of the laser mode is required at startup, in order to drive the controller in the proper direction. This condition is normally satisfied, unless service work has been performed.

5.7 Summary

In an effort to further improve the performance of the automatic mode control system, experiments were initiated to find a means of controlling the second primary resonator mirror. Previous experience by researchers in this laboratory and others, has shown that it is difficult, even for a human operator, to properly align both primary mirrors in an unstable resonator system, based only on visual observations of the beam's uniformity. To overcome this difficulty, the beam steering property of optical resonators was employed, to provide a second measure of the resonator's alignment error.

To monitor the pointing angle of the output beam, modifications were made to the beam profile sensor and its interface circuitry, to obtain 2 new error signals. These signals supplied measures of the beam's horizontal and vertical position on the sensor array. The responsivity of these quantities was subsequently measured, and used to complete the control system model. Based on this model, a relay controller was designed and tested. Finally, experimental data was collected, verifying the ability of the completed system to perform simultaneous stabilization of mode uniformity, pointing angle, and output power level.

The data collected during these tests revealed that the system was capable of establishing proper resonator alignment, and chosen power level, from a cold start condition, within a maximum of approximately 30 seconds. Furthermore, visual observations indicated that the system provided a consistent alignment between different test runs. This would be especially beneficial in industrial systems, where consistency between production runs is important. In addition to the quick start capability of the system, testing revealed that continuous alignment of the resonator during extended runs significantly reduced variations in pointing angle. This reduction in pointing angle drift was most dramatic during the initial 10 to 15 minute warm-up period.

During the course of the experiments with the 5 kW PIE-3 laser system, a 20 kW $C0_2$ laser system became available in the laboratory. It was decided to take advantage of this opportunity, to construct and test a prototype controller for the higher power laser. The following chapter provides details of the design, installation, and testing of this system.

CHAPTER 6

PROTOTYPE CONTROLLER DESIGN

Up to this point in the study, testing of the mode control system was performed exclusively on the 5 kW, PIE-3 laser. This system provided a convenient means for testing the uniformity, position and power level control loops. However, industry's desire for higher penetration depths and processing rates has resulted in the need for lasers capable of producing a stable output mode at power levels much higher than 5 kW. Fortunately in this regard, during the course of these experiments with the PIE-3 system, a new laser, capable of power output in excess of 25 kW[1], was added to our laboratory's inventory of test equipment.[54] This machine thus provided the opportunity to test the performance of the automatic mode control system, at higher power levels, and under conditions similar to those encountered in actual industrial welding and cutting applications.

The next section briefly discusses the optical and power control systems used in the new laser. The subsequent sections provide details of a prototype mode control system, designed for permanent installation on this machine. First, the beam profile sensor was reconfigured to match the PIE-4 laser's larger beam diameter. Printed circuit boards were then designed to accommodate the interfacing and alignment control circuits. At the same time, a graphical interface was added to the microcomputer based data acquisition system. This modification enhanced the diagnostic capability of the system by providing on-line display of various signals from the pyroelectric detector array and other sensors. After installation of the hardwired controller and the data-acquisition system was complete, a new mathematical model was constructed to represent the new system. The power control portion of the system was redesigned to take advantage of the laser's continuously variable, silicon controlled rectifier (SCR) type power supply. Modifications were also made to the beam position controller in order to minimize its effect upon mode uniformity. Finally, data was collected during several closed-loop runs to evaluate the transient and steady-state performance of the completed system.

---

[1] This laser has been dubbed "PIE-4" among the laboratory staff.

6.1 Test Laser

The PIE-4 laser system is very similar in structure to the PIE-3 laser, described earlier. However, several design modifications were implemented during the construction of this machine. Many of these have been reported elsewhere, so only those which directly concern the present research are discussed here.[54]

6.1.1 Optical System

The optical resonator, shown in Fig. 6.1, is of the same configuration as that used in the PIE-3 laser. Resonator parameters are slightly different, with a resonator length of 9.1 m, and mirror curvatures of -44.65 m and 40.5 m, for M1 and M2, respectively. In addition, 2 changes of special interest were made to the physical construction of the optical system.

Specifically, the entire resonator assembly was mechanically isolated from the laser vessel by mounting it on a spring mounted, rubber damped, optical bench. In an effort to minimize the effects of high operating temperatures, this entire assembly was equipped with a temperature controlled, water cooling system. At the time this laser was designed, this arrangement represented the state of the art technique for minimizing thermally induced alignment drift.

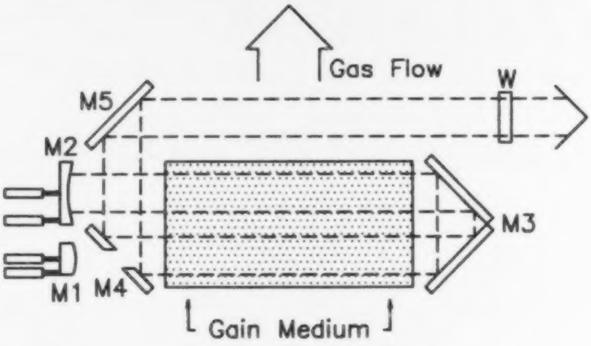


Fig. 6.1 Folded Path Laser Resonator

The resonator mirror mounts were also redesigned, to permit the use of standard motor-micrometers. As such, adapting the alignment system to achieve automatic mode control was simplified considerably. The existing motor-micrometers were simply replaced with digital readout units. Although the digital readout is not required for operation of the alignment controller, the position information supplied by these micrometers was convenient for the modeling procedure.

6.1.2 Power Supply

Important changes were also made to the DC power supply. As discussed in Chapter 3, discharge current to the PIE-3 laser was set using a motor-driven variac. This variac allowed changes in the DC excitation current at a fixed rate. Consequently, the only feasible power controller structure was a non-linear, relay type. The PIE-4 laser however, is equipped with a variable duty cycle, silicon controlled rectifier (SCR) type power supply. This power supply allows the laser's discharge current to be varied continuously, and a feedback controller maintains the current at the selected level. The addition of an auxiliary setpoint input was the only modification required to achieve feedback control of the laser's output power.

Output power measurements were performed by a combination power meter/beam dump, already installed on the laser.[54] This device, although it does not provide on-line power measurement, was adequate for the present investigation. It absorbs all of the output laser radiation, and provides an accurate means for measuring the average power output. A 0-5 V output, linearly proportional to beam intensity, was available to the data-acquisition and power control systems.

Discharge current was measured using a Hall-probe; discharge voltage was monitored with a resistive voltage divider. Both of these circuits were already installed on the laser, and were equipped with 0-5 V outputs.

6.2 Hardwired Controller Installation

6.2.1 System Overview

A block diagram of the PIE-4 mode controller installation, appears in Fig. 6.2. The rotating wand sampling stage deflects 0.25 % of the laser's output

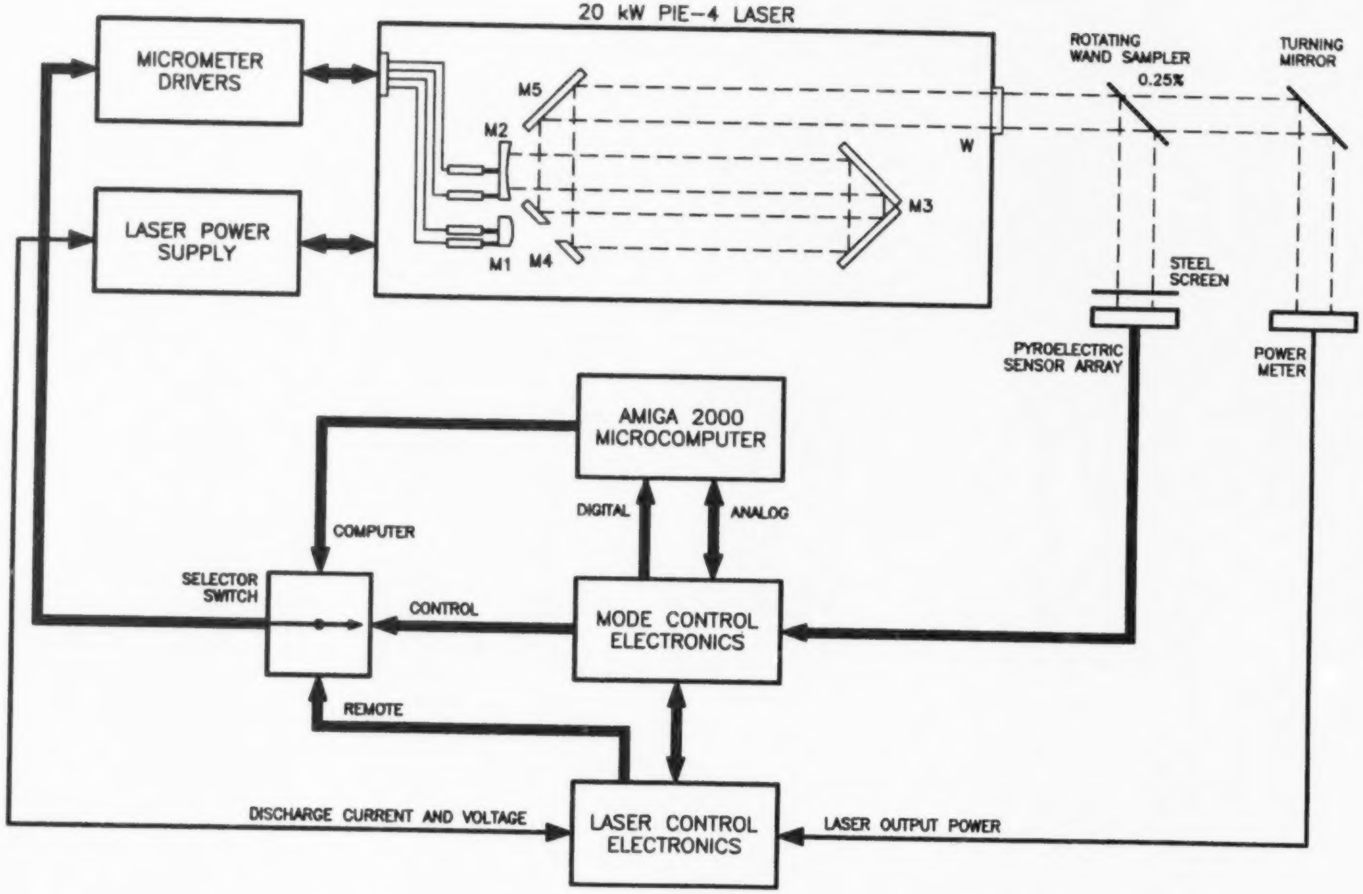


Fig. 6.2 Block Diagram of the PIE-4 Mode Controller Installation

onto the pyroelectric sensor array. A metal screen, placed in front of the array was used to spatially smooth and attenuate the sample[2]. The interfacing and control electronics are housed in a single cardrack. A number of cables link these circuits to the micrometer drivers, the microcomputer and the existing laser electronics. A 3-position switch determines whether the motor-micrometers are controlled manually, by the mode controller, or by the computer. As can be seen in Fig. 6.2, both primary mirrors may be remotely aligned using these motor-micrometers.

6.2.2 Alignment Servo Installation

As mentioned previously, the mirror mounts in the PIE-4 laser were redesigned to permit direct drive by motor-micrometers. For these experiments, the existing motormics on both primary mirrors were replaced with digital feedback units.[55] This was to allow the collection of precise servo position data, for use in the construction of the system model. Unfortunately, the load placed on these relatively low torque units was still heavy enough to prevent reliable operation at slower speeds. The best overall performance was obtained by continuing to use a relay controller, with the micrometer speeds preset to fixed values.

The new optical system was entirely enclosed in the laser vacuum enclosure. Although this eliminated the need for hermetic seals around the micrometer shafts, it placed the motormics, and the accompanying digital circuitry, in close proximity to the high voltage discharge. Careful attention to cable shielding and routing was required in order to minimize electrical interference.

6.2.3 Sensor Array Installation

Initially, the sensor described in Chapter 4, was installed on the laser. Preliminary tests verified that it detected mode nonuniformity properly. However, the position error signals did not respond as well as expected. Subsequent investigation revealed that the sensors were not appropriately placed to detect position shifts of the somewhat larger annular beam produced by the PIE-4 laser.

---

[2] This technique was first used with the original, thermistor based sensor array. The screen helps provide a better indication of mode uniformity by slightly diffusing the beam sample, before it strikes the sensor array. It was also observed that higher laser power levels could be accomodated by utilizing additional screens.

To solve the problem, a new sensor array board was constructed, configured as shown in Fig. 6.3. As illustrated in the figure, the rings are more tightly spaced than in the original unit, and the number of detectors is reduced from 56, to 32. This decreased its cost by approximately 40%, without reducing performance. A simple x-y mount allowed accurate placement of the sensor array.

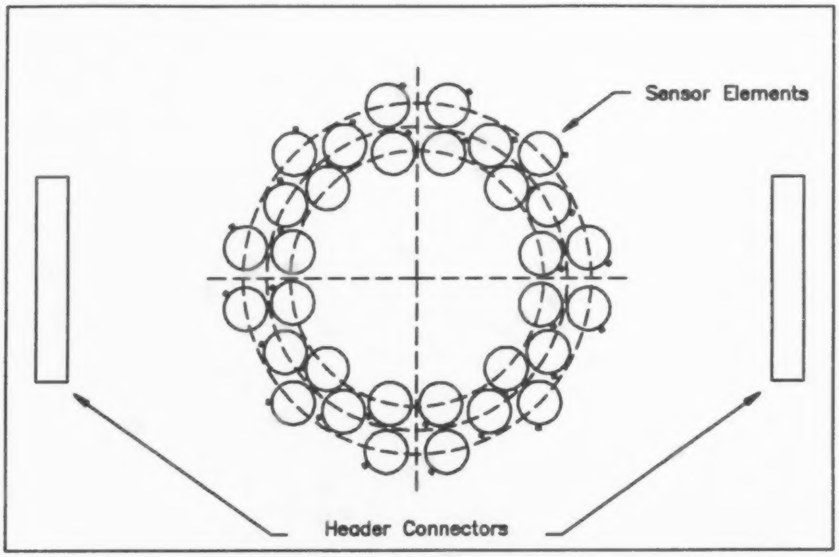


Fig. 6.3 PIE-4 Beam Profile Sensor Array

6.2.4 Controller Circuit Boards

All of the PIE-4 laser control electronics are mounted in a commercial, electronics rack. An unused slot in this rack supplied a convenient location for installation of the mode controller circuitry. A number of printed circuit boards were designed and fabricated to duplicate the functions performed by the breadboarded PIE-3 electronics.

Fig. 6.4 illustrates the subdivision of the controller and interface circuitry among the 8 circuit boards. A low voltage DC power supply provides ±15 and +5 volts to each board. Interfacing circuitry is subdivided into four separate boards, one for each quadrant of the sensor array. Outputs from these boards are fed to the uniformity and position control boards. These two boards calculate the error

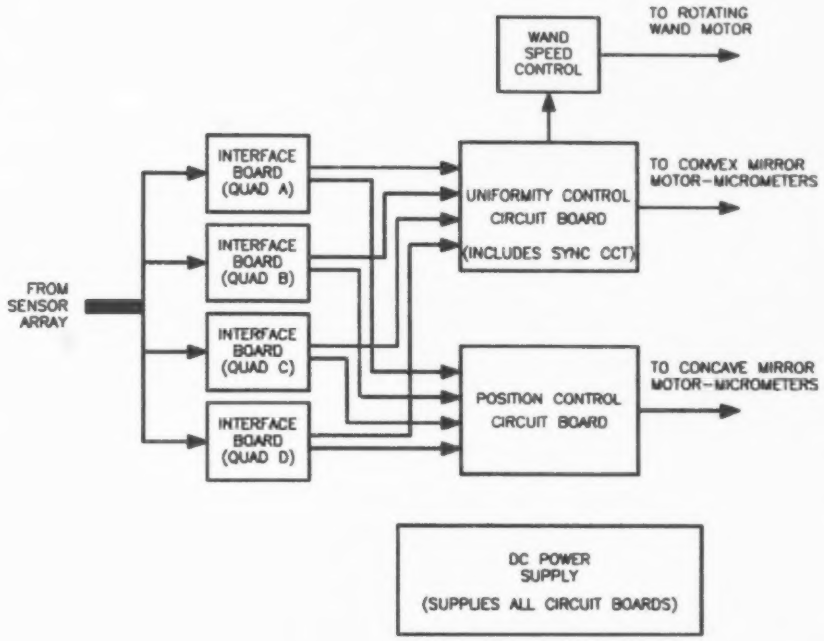

WAND SPEED CONTROL
TO ROTATING WAND MOTOR
INTERFACE BOARD (QUAD A)
INTERFACE BOARD (QUAD B)
INTERFACE BOARD (QUAD C)
INTERFACE BOARD (QUAD D)
UNIFORMITY CONTROL CIRCUIT BOARD (INCLUDES SYNC CCT)
TO CONVEX MIRROR MOTOR-MICROMETERS
FROM SENSOR ARRAY
POSITION CONTROL CIRCUIT BOARD
TO CONCAVE MIRROR MOTOR-MICROMETERS
DC POWER SUPPLY
(SUPPLIES ALL CIRCUIT BOARDS)

Fig. 6.4 Arrangement of the Mode Controller Circuit Boards

signals and activate the alignment servos, as needed, to stabilize beam uniformity and position. The wand synchronization circuit, which controls sample timing, is also mounted on the uniformity control board.

During studies with the PIE-3 laser the speed of the sampling wand was adjusted manually using a variac. Although this approach was satisfactory for those experiments, it was found that the rotation speed of the wand varied with temperature. This led to corresponding changes in the data sampling period. Although this did not adversely affect the performance of the control system, it complicated digital filtering of the data. To rectify this problem, and to eliminate the variac, a solid state AC motor driver was constructed. A simple, proportional speed controller was employed to control this driver circuit so as to maintain a constant sampling interval of 0.1 second. One circuit board sufficed to accommodate the speed controller, and solid state motor driver circuits

6.3 Microcomputer Installation

6.3.1 Hardware

Data collection was performed using the same Amiga microcomputer employed in the PIE-3 experiments. As before, a PROTO-40K data-acquisition board was used to monitor signals from the mode controller and laser electronics. Analog data was collected from each of the 12 ring segments in the sensor array, and from the laser's power, discharge current, and voltage monitors. The digital servo control signals produced by the electronic mode controller circuits were fed to the digital inputs of the data-acquisition board.

The motor-micrometers could be controlled by the computer, using a special micrometer control circuit board, designed to operate in conjunction with the Proto-40K board. A digital counter on this board was employed to monitor the micrometer positions. Depending on the setting of the source selector switch, shown in Fig. 6.2, servo movements could either be directed by the hardwired controller, or the signals could be intercepted and modified as desired, under program control.

6.3.2 Software

Several modifications were made to the data-acquisition and control software, to increase system performance, and to enhance its usefulness and ease of operation. First, a graphical interface, shown in Fig. 6.5, was designed. This interface provides the user with on-line data display, and a collection of mouse gadgets to control various program features.[3] Real time data display simplifies controller tuning, and provides continuous monitoring of the laser's alignment quality, and output power level. This capability is especially useful during material processing operations, such as cutting and welding. While performing these types of processes, the only evidence of laser operation is normally a shower of sparks, or the completed weld. As shown in Fig. 6.5, laser power, discharge current and voltage, and discharge efficiency are displayed numerically. Two timers are also

---

[3] A more detailed description of the operational features of this system is provided in the appendices.

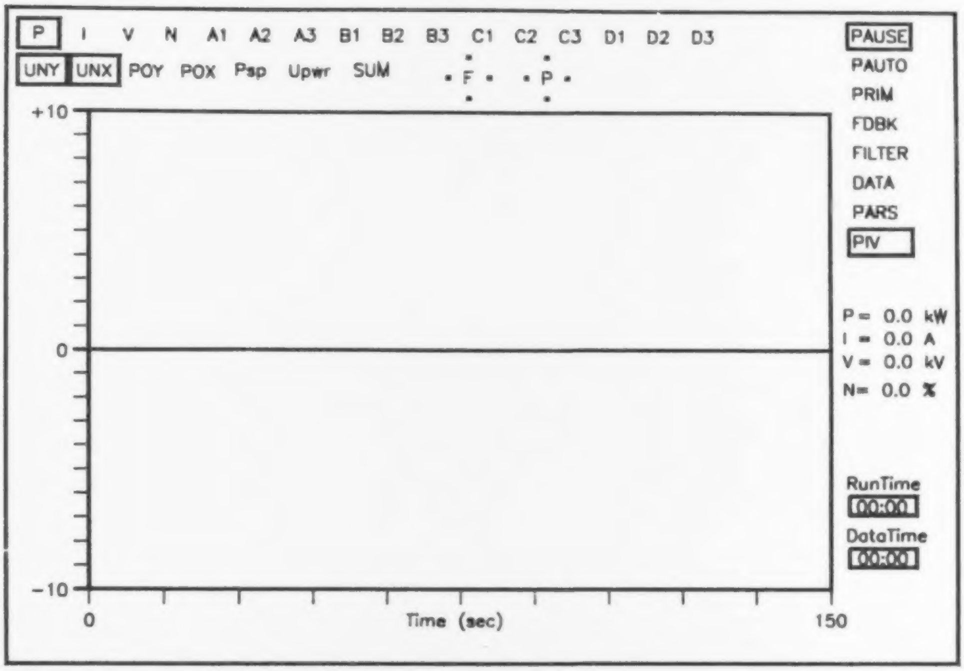


Fig. 6.5 Data Acquisition System Graphical Display

provided on the display. The upper readout displays the elapsed runtime while the lower indicates how much time has passed since data collection was enabled. Centered over the plotting area[4] are two x-y graphical displays which indicate the direction of alignment servo movements.

As mentioned earlier, the PIE-4 laser was equipped with a silicon controlled rectifier type power supply. To fully utilize this capability, a proportional-integral (PI) controller was implemented in software. A software implementation supplied an ideal means of testing system performance, without the expense and complication of additional hardware. A mouse gadget enables or disables automatic power control.

After installation of the system hardware was complete, a few test runs were performed, to obtain an initial indication of the controllers performance with the new laser. It soon became apparent that the position control portion of the system did not operate effectively. This was because the direct drive arrangement, used in the new resonator mirror mounts did not permit alignment adjustments to the concave mirror to be performed slowly enough. Large disturbances in beam uniformity were observed as the position controller adjusted the concave mirror. One possible solution to the problem would have been to redesign the controller electronics to permit driving the micrometers with pulses. A more flexible approach, however, was to employ the Amiga microcomputer in place of the hardwired controller. Using the computer, various control strategies were investigated, simply by modifying the program code[5].

---

[4] The letters "F" and "P" were used to indicate "Feedback" and "Primary" mirror. This is consistent with the terminology commonly employed by the laboratory staff (convex=feedback, concave=primary).

6.4 System Model

In Chapter 5, the mode controller was shown to be capable of stabilizing both beam uniformity and position. At that stage in the experiments, this was accomplished by aligning the concave mirror to compensate for beam position drifts. The convex mirror was employed to continuously maintain mode uniformity. Although this method provided acceptable performance, it was observed that large disturbances to mode uniformity resulted when similar micrometer speed settings were chosen in both mirrors.

Closer examination revealed that, whenever the position controller initiated alignment adjustments to the concave mirror, the uniformity controller invariably made adjustments to the convex mirror to compensate. These observations suggested that feed-forward compensation might improve the performance of the position controller. It was hoped that, by moving both mirrors simultaneously, beam position adjustments could be performed with less effect upon mode uniformity.

The new system model which reflects this approach is shown in Fig. 6.6. Two changes were made to the alignment control loops. The uniformity and position systems are no longer independent of each other. Rather, the system is now divided into two independent systems, one for each of the horizontal and vertical axes. In order to implement feed-forward compensation, it was necessary to model the sensitivity of beam uniformity to changes in concave mirror alignment. This is represented in Fig. 6.6 by two additional parameters, Kyu2 and

---

[5] A more detailed description of the resulting non-linear controller is provided later in this chapter.

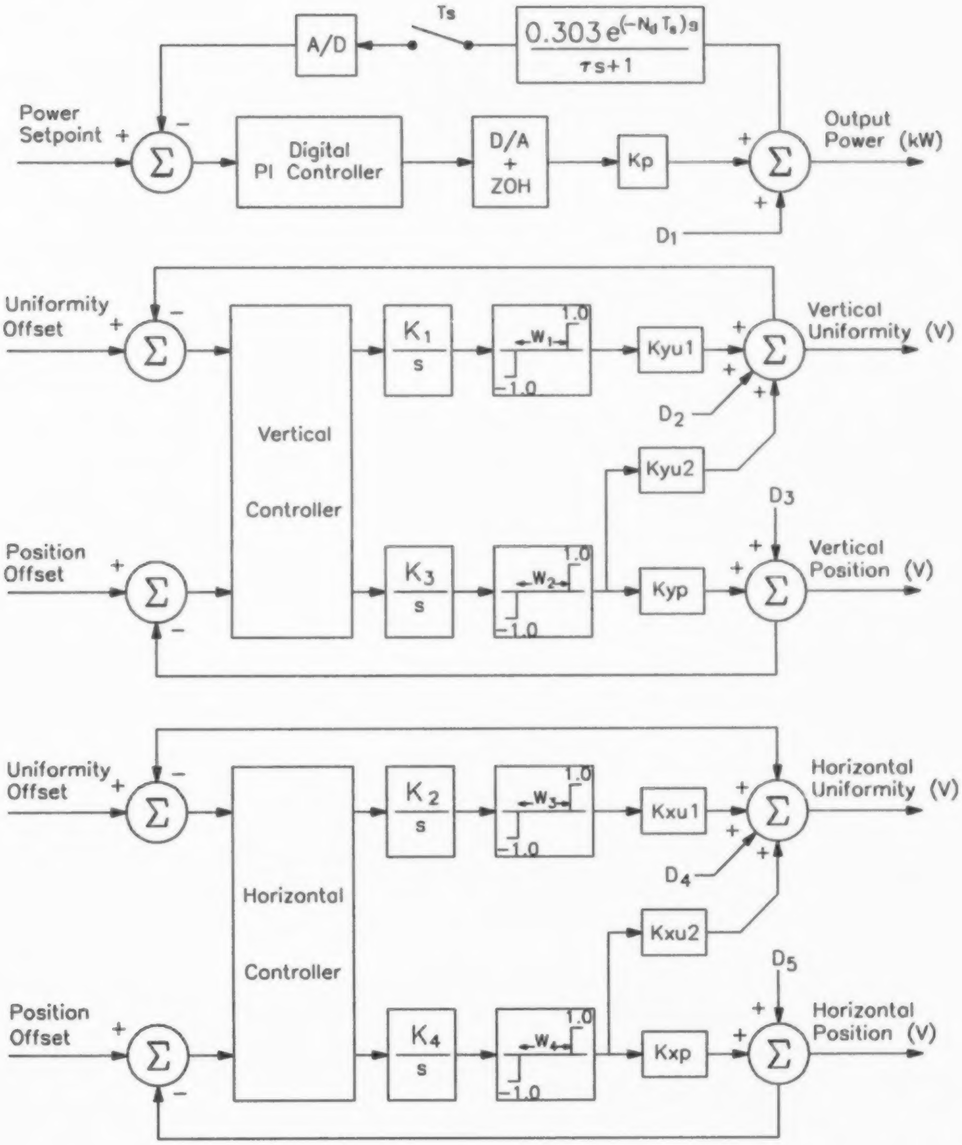


Fig. 6.6 Mathematical Model for the PIE-4 Mode Control System

Kxu2 (V/μ). Also, a backlash block was introduced into each of the loops, to account for the micrometer gear coupling. The backlash in these servos had previously been ignored, but has now been included for completeness.

The structure of the power control portion of the system is also changed considerably. A single gain factor, Kp, represents the sensitivity of the laser output power to deviations in the discharge current controller setpoint. In the feedback path, the power meter response is modeled by a first order block, with a time delay component. The gain of this unit (0.303 V/kW) was previously known. This power meter block is followed by a sampling stage, and an analog to digital converter. The loop controller is a conventional, proportional-integral type[6]. A digital to analog converter with a zero order hold, transfers the controller output to the setpoint of the laser power supply.

---

[6] This controller is implemented as a subroutine in the computer code. Additional detail is provided later in this chapter.

6.4.1 Alignment System Parameter Estimation

Data to estimate the uniformity portion of the system were collected automatically, using a simple subroutine. After aligning each mirror along the axis under test, the micrometer was jogged about the aligned position while data were recorded. This procedure was repeated on each of the 4 alignment servos. Estimates of the uniformity sensitivity factors, and of the backlash of each micrometer, were extracted from this data using the simulation of Fig. 6.7, programmed on TUTSIM. This simple simulation effectively compares the measured data to the output generated by the model. The square of the error between the measured and model outputs is integrated to provide a suitable measure for evaluating the quality of the fit between the model and the real data. The parameters to be estimated are then iteratively adjusted by the TUTSIM program to minimize this error measure. Comparisons between the estimated and measured responses for each micrometer, are shown in Figs. 6.8 and 6.9.

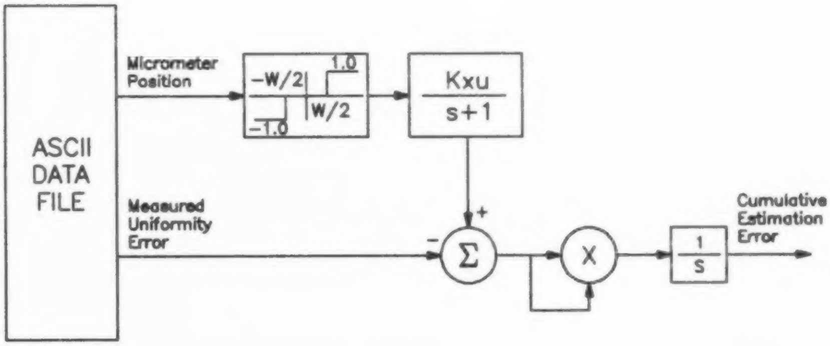


Fig. 6.7 TUTSIM Model used to Estimate the Uniformity System Parameters

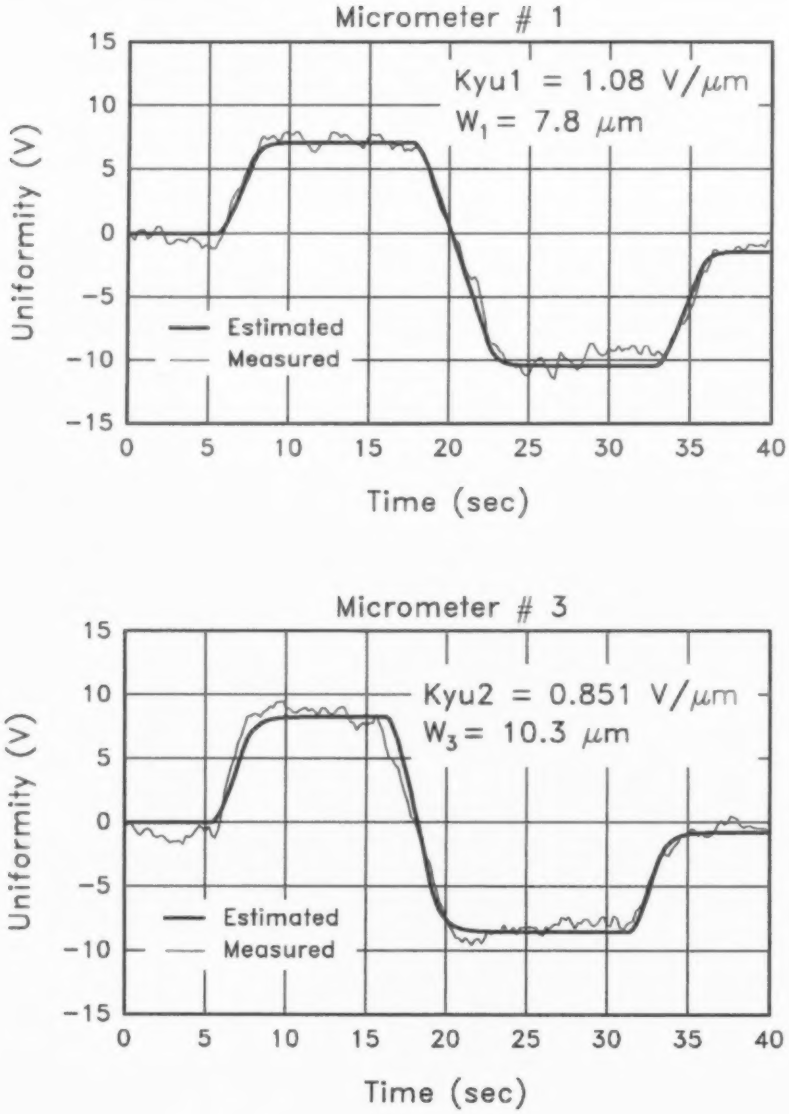


Fig. 6.8 Estimation of the Vertical Uniformity and Backlash

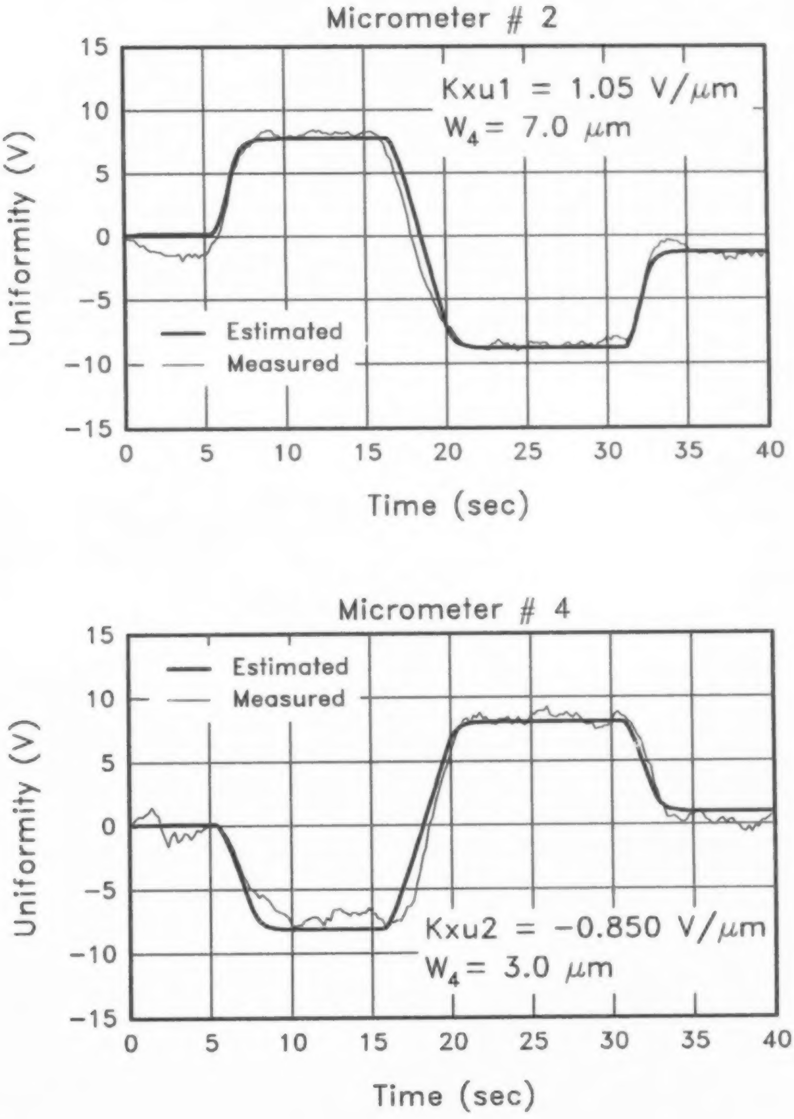


Fig. 6.9 Estimation of the Horizontal Uniformity and Backlash

The parameters K_{yp} and K_{xp}, in the position control loop, were estimated from data recorded at equally spaced intervals, about the center position. After each adjustment to the concave mirror micrometer, the uniformity controller was allowed to align the convex mirror. A number of samples were then collected and averaged at each position. As shown in Fig. 6.10, a simple first order fit supplied the vertical and horizontal position sensitivity factors.

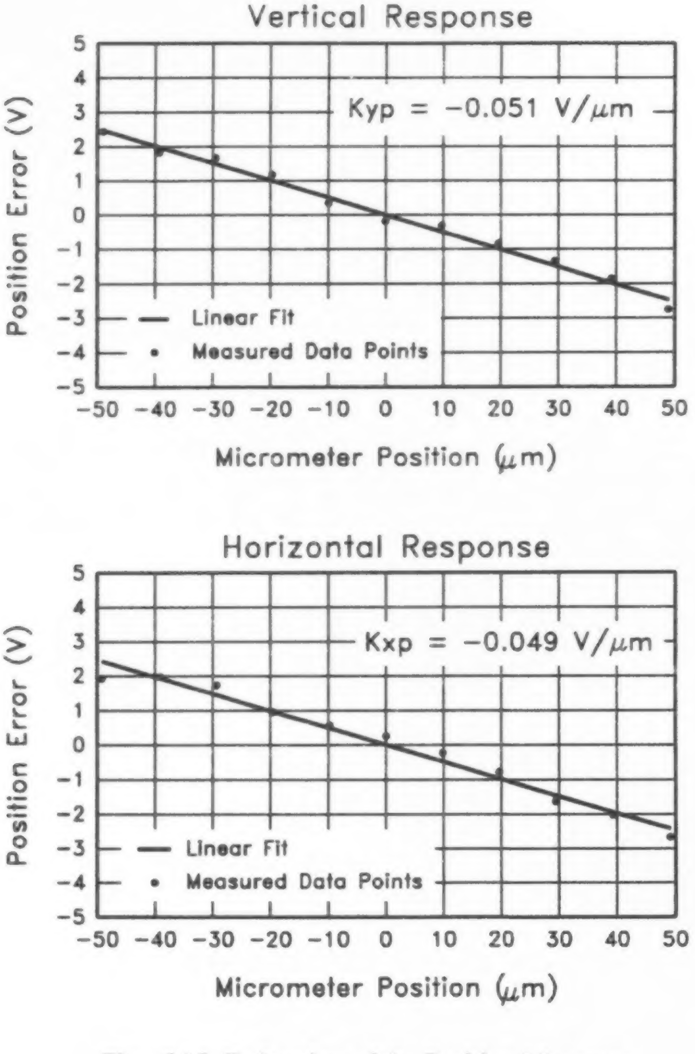


Fig. 6.10 Estimation of the Position Measures

6.4.2 Power Control System Parameter Estimation

Data to estimate the combined laser and power meter response to changes in discharge current was recorded using a standard step response procedure. A suitable steady-state power level was selected, and the current controller setpoint was then perturbed about the center value. The batch least squares method was used to fit a discrete model to this data as follows:

$$P_o(n) = a_1\, P_o(n-1) + b_1\, V_S(n-1-N_d) \tag{6.1}$$

where: P_o = Laser output power
V_s = Current controller setpoint
a_1, b_1 = parameters to estimate
N_d = power meter delay in sample periods
n = sample number

A least squares estimate of the parameters a_1 and b_1 is given by:

$$\Theta = (\Phi^T \Phi)^{-1}\, \Phi^T\, \mathbf{P}_o \tag{6.2}$$

where:

$$\Theta = \begin{vmatrix} a_1 \\ b_1 \end{vmatrix} \qquad \mathbf{P}_o = \begin{vmatrix} P_o(N_d+2) \\ P_o(N_d+3) \\ P_o(N_d+4) \\ \cdot \\ \cdot \\ \cdot \\ P_o(N) \end{vmatrix} \qquad \Phi = \begin{vmatrix} P_o(N_d+1) & V_s(1) \\ P_o(N_d+2) & V_s(2) \\ P_o(N_d+3) & V_s(3) \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ P_o(N-1) & V_s(N-1-N_d) \end{vmatrix}$$

N = the number of samples in the I/O vector

The delay, N_d, was determined as six sample periods from graphical examination of the data. The MATLAB program was used to calculate Θ, and the resulting discrete model was converted to the continuous domain using:

$$\tau = \frac{-T_s}{\ln(a_1)} \qquad (6.3)$$

$$K_p = \frac{b_1}{1 - e^{-(T_s/\tau)}} \qquad (6.4)$$

where: T_s = sampling period (0.217 seconds)

The output from this model is plotted in Fig. 6.11 together with the measured data and current controller setpoint. It can be seen that the model response closely matches the power meter output. The graph also clearly reveals the 1.3 second delay in the power meter response. A block diagram of the completed PIE-4 system model, complete with estimated parameters, appears in Fig. 6.12.

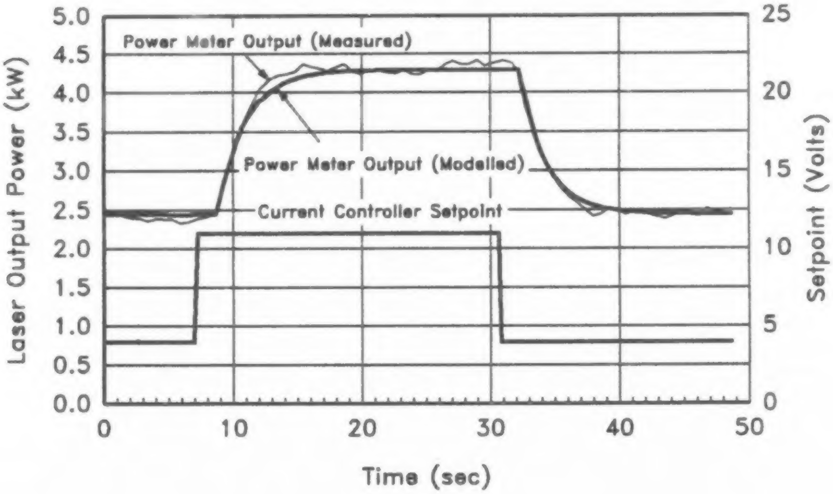


Fig. 6.11 Estimation of the PIE-4 Power Control System

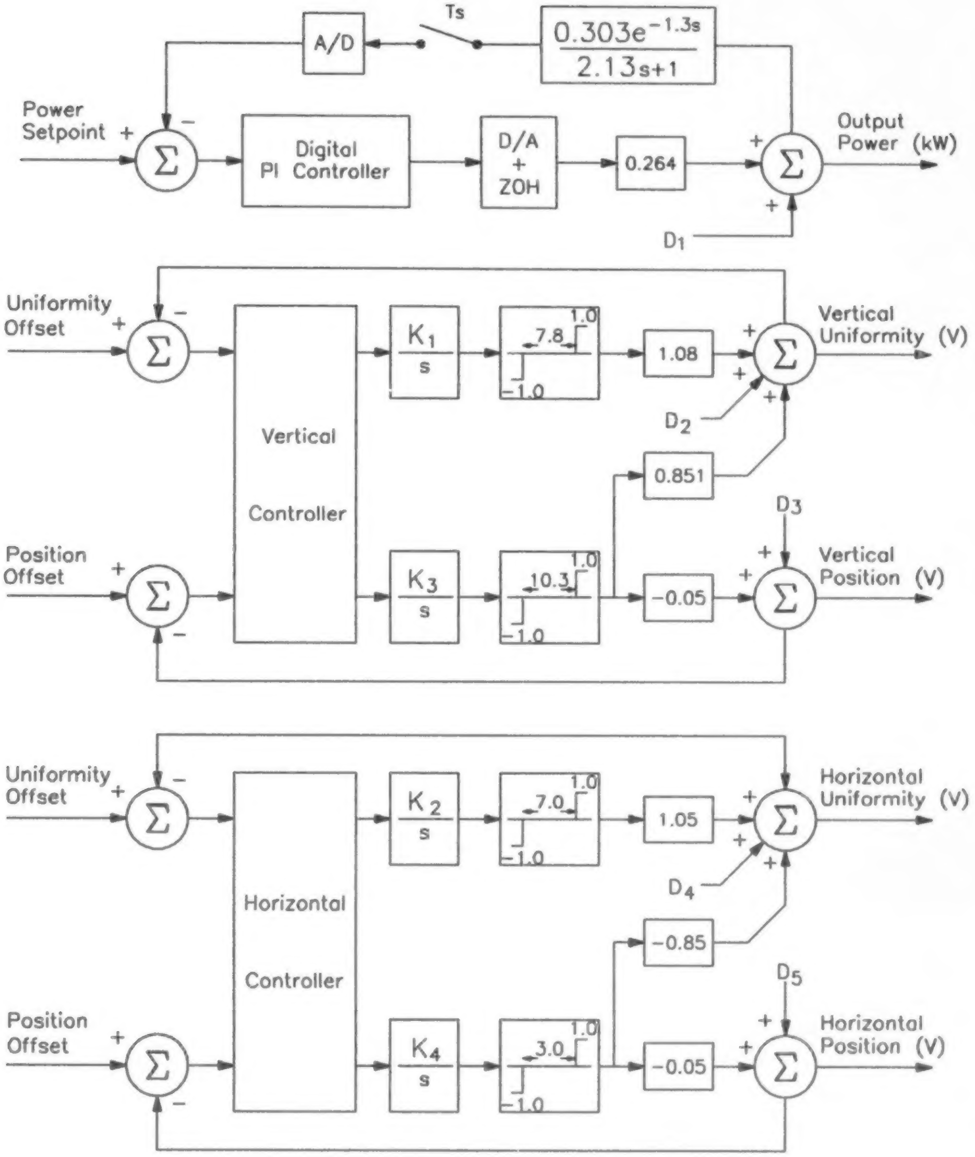


Fig. 6.12 Completed PIE-4 System Model

6.5 Controller Design

As discussed previously, the position controller provided poor results during initial testing. At that point, it was decided that the final experiments should be performed using the computer in place of the hardwired controller. This approach was considerably more flexible, and did not require the design of new electronics.

A single program was written to perform all of the data-acquisition and control functions. The following sections describe the various components of this program.

6.5.1 Control Program Structure

A simplified flowchart of the control/data-acquisition program is shown in Fig. 6.13 [7]. After the various initialization requirements have been attended to, the program enters the main loop where the data collection and control functions are performed. Execution of the loop begins upon detection of the synchronizing signal from the rotating wand, and is completed within 1 sample period.

Immediately upon entry to the loop, a subroutine checks for user input; if any mouse activity has occurred, the appropriate flag or variable is modified accordingly. Next, the raw analog and digital data is read from the PROTO-40K board. If the user has enabled data storage, the data is stored in ASCII format, in a RAM data file. The following block calculates the beam position and output power level error signals, and then applies lowpass digital filtering to them before they are passed on to the control subroutines. If the user has selected the "FILT" mouse gadget, the raw data is also digitally filtered. The power control block implements a standard, digital PI controller. It calculates a control signal and sends it to the laser's power supply setpoint, via the PROTO-40K digital to analog converter. The alignment control block performs a similar function for the resonator alignment portion of the system. This routine directs the movements of the 4 micrometers as necessary to stabilize beam uniformity and position. The last subroutine in the loop

---

[7] This flowchart provides only a general overview of the program structure. The operation of the power and alignment controller blocks is discussed in more detail in later sections. Complete program listings appear in the appendices.

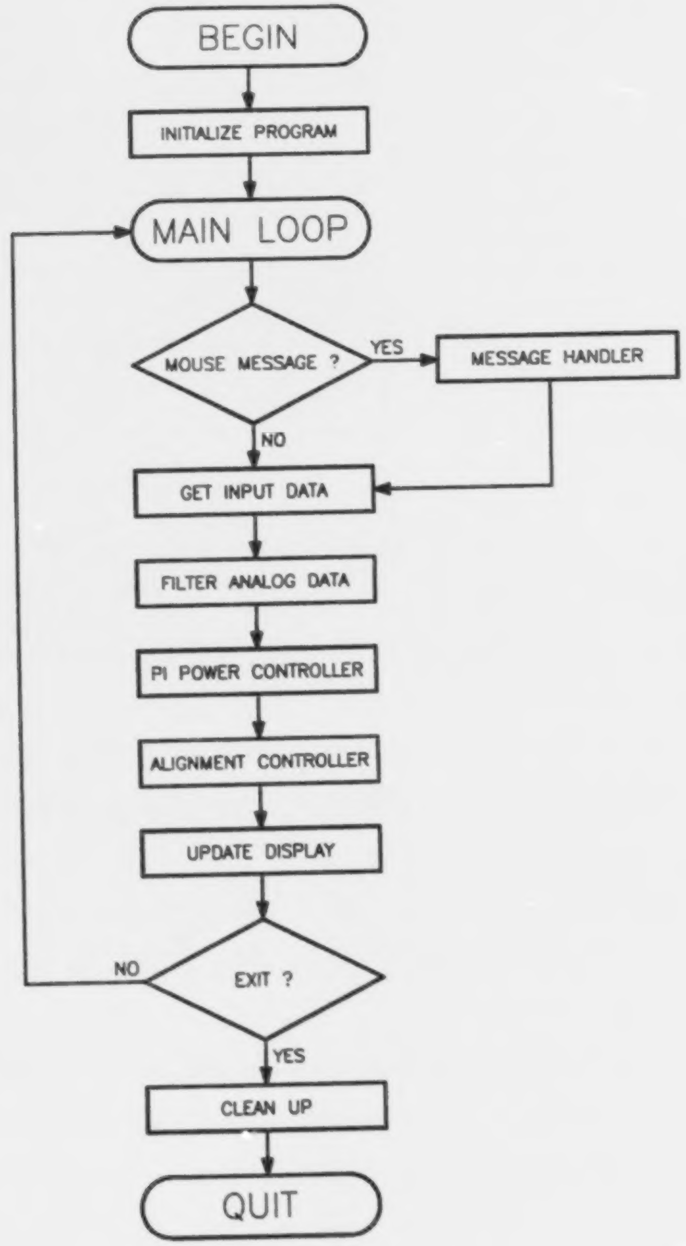


Fig. 6.13 The Data-Acquisition and Control Program

services the graphical interface display. This includes plotting the analog data signals, and updating the timers and other numerical displays. Finally, if the user has selected to end the session, execution of the loop is terminated. After completing a few clean-up chores, the program exits.

6.5.2 Proportional-Integral Power Controller

A continuous time PI controller is often written in the form:

$$u(t) = K_c \left[e(t) + \frac{1}{\tau_i} \int e(t)\ dt \right] \qquad (6.5)$$

where: K_c = the controller gain
τ_i = integral time
$u(t)$ = controller output
$e(t)$ = input error signal

One common method used to obtain the discrete equivalent of equation 6.5 is to approximate the integration term by a rectangular integration.[56] The discrete PI control algorithm then takes the form:

$$u(n) = K_c \left[e(n) + \frac{T_s}{\tau_i} \sum_{i=0}^{n} e(i) \right] \qquad (6.6)$$

where: K_c = the controller gain
τ_i = integral time
$u(n)$ = controller output during nth sampling period
$e(n)$ = input error signal during nth sampling period
T_s = sampling period

This form is referred to as the positional form of the PI controller because the total output of the controller is calculated. A more useful incremental form may be derived if a change in control signal is defined as:

$$\Delta u(n) = u(n) - u(n-1) \qquad (6.7)$$

which leads to:

$$\Delta u(n) = K_c \left[e(n) - e(n-1) + \frac{T_s}{\tau_i}\, e(n) \right] \qquad (6.8)$$

Equation 6.8 affords two operational advantages over the positional form. It is a recursive algorithm which is more amenable to computer implementation, and it does not require initialization for bumpless transfer from manual to automatic control. Consequently, this form was selected and a relatively simple subroutine was written to implement the controller.

6.5.3 Uniformity and Position Controller

As discussed previously, it was thought that improved performance of the alignment controller could be obtained by using feed-forward compensation. To visualize the feed-forward concept, consider as an example the vertical portion of the alignment system. As can be seen in Fig. 6.12, beam uniformity is almost equally sensitive to movements of either micrometer. Suppose one were to move micrometer #3 at speed K3 (μm/s), while moving #1 in the opposite direction at a speed determined by:

$$K_1 = \left(\frac{0.851}{1.08} \right) K_3 \qquad (6.9)$$

Under these conditions, it should be possible to alter the beam position without disturbing mode uniformity. In order to implement this control action, a hybrid system was developed. This design utilized the electronic uniformity controller in conjunction with a software position controller.

The controller's principle of operation is straightforward. Consider, once again, the vertical portion of the alignment system in Fig. 6.12. Every 0.1 s, the program reads the control signals from the electronic uniformity controller, and then transfers them to the convex mirror micrometers. Occasionally, the routine disables uniformity control, and if necessary, simultaneously moves both micrometers a short distance in the appropriate directions, to correct any position error which may be present. When this movement is completed, control of the convex mirror micrometers is immediately returned to the uniformity controller.

When the system is at, or near, steady-state, the routine checks and adjusts beam position every five seconds. If the position error is large, such as at startup, the interval between control movements will decrease to as little as 1 second. The pulse length used by the position controller was chosen experimentally to obtain the fastest possible response, without producing uniformity disturbances.

6.6 Experimental Results

Tuning of the alignment and power controllers was simplified considerably by the on-line, data display capability of the system. After satisfactory tuning parameters were selected, a few final runs were performed to document the transient and steady-state performance of the new power and alignment controllers.

6.6.1 Power Controller

Based on the power meter's effective low-pass cutoff frequency of 0.075 Hz, a sampling frequency of 2 Hz was chosen for the power control loop. When tuning the power controller it was necessary to allow for the slow response of the laser power meter. Because the output from the laser responds immediately to changes in discharge current, it is more appropriate to observe input current, rather than output power during the tuning procedure. This method ensures that the laser's output power does not rise to dangerous levels before the power meter responds. The controller parameters were finally selected as Kc = 1.25, and τ_i = 1.5 seconds. Final evaluation of the controller response was performed by subjecting the system to a setpoint change from 3 kW up to 5 kW, and then back down to 3 kW. The laser's output power and discharge current are plotted in Fig. 6.14.

6.6.2 Alignment Controller

Tuning of the alignment controller was performed by first tuning the uniformity portion of the system. Initially, the uniformity controller gain was adjusted to the maximum level at which the uniformity error signals would remain inside the ± 0.6 V control window, under steady-state conditions. After selecting values of 0.35, for both the vertical and horizontal directions, the speeds of the two convex mirror micrometers were tuned to obtain a satisfactory response. Speed

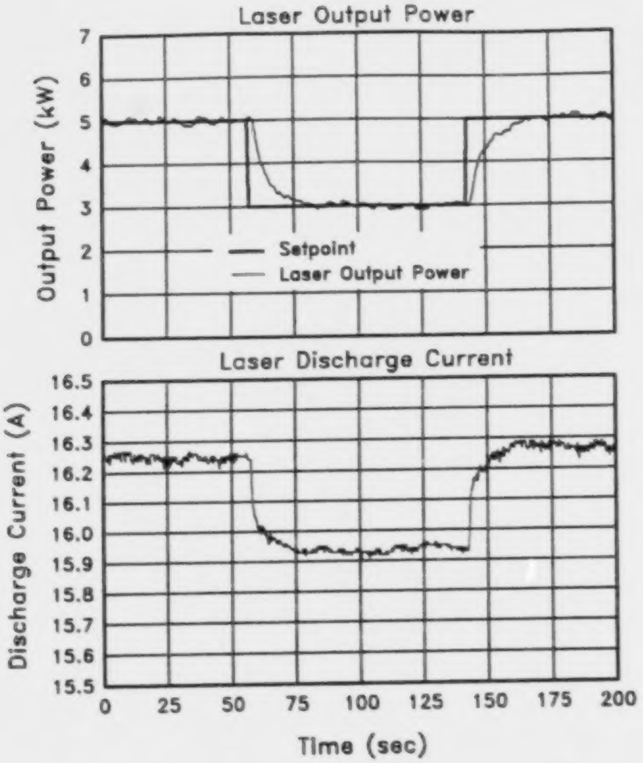


Fig. 6.14 Power Controller Step Response

settings of 5.0 μm/s for micrometers 1 and 2, were found to give fast response with no overshoot.

With the uniformity controller fully operational, it remained only to tune the position portion of the system. Once again, the gains were adjusted as high as possible, while still remaining inside the control window. Gains of 1.5 were selected for both axes. The speeds of micrometers 3 and 4 were then set at 6.3 μm/s, as determined by equation 6.9.

The ability of the alignment system to recover from a severe misalignment was tested by misaligning the concave mirror by 50 μm about each axis. While the micrometers were moved, the uniformity controller remained on automatic to maintain a uniform mode as the beam position was shifted off center. The uniformity controller was subsequently disengaged, and the convex mirror was misaligned by 20 μm about each axis.[8] After allowing the error signals to settle, the uniformity controller was reengaged. When the uniformity error signals returned to within the control window, the position controller was also set to automatic. The uniformity and position measures are plotted in Fig. 6.15, along with (for reference) the status of the manual/automatic switches for each controller.

---

[8] For the concave mirror, a misalignment of 50 μm (65 urad) was chosen to obtain a significant visual shift in beam position. In the case of the convex mirror, 20 μm (16 urad) was approximately determined as the misalignment required to reduce laser output to 1/2 of its original intensity.

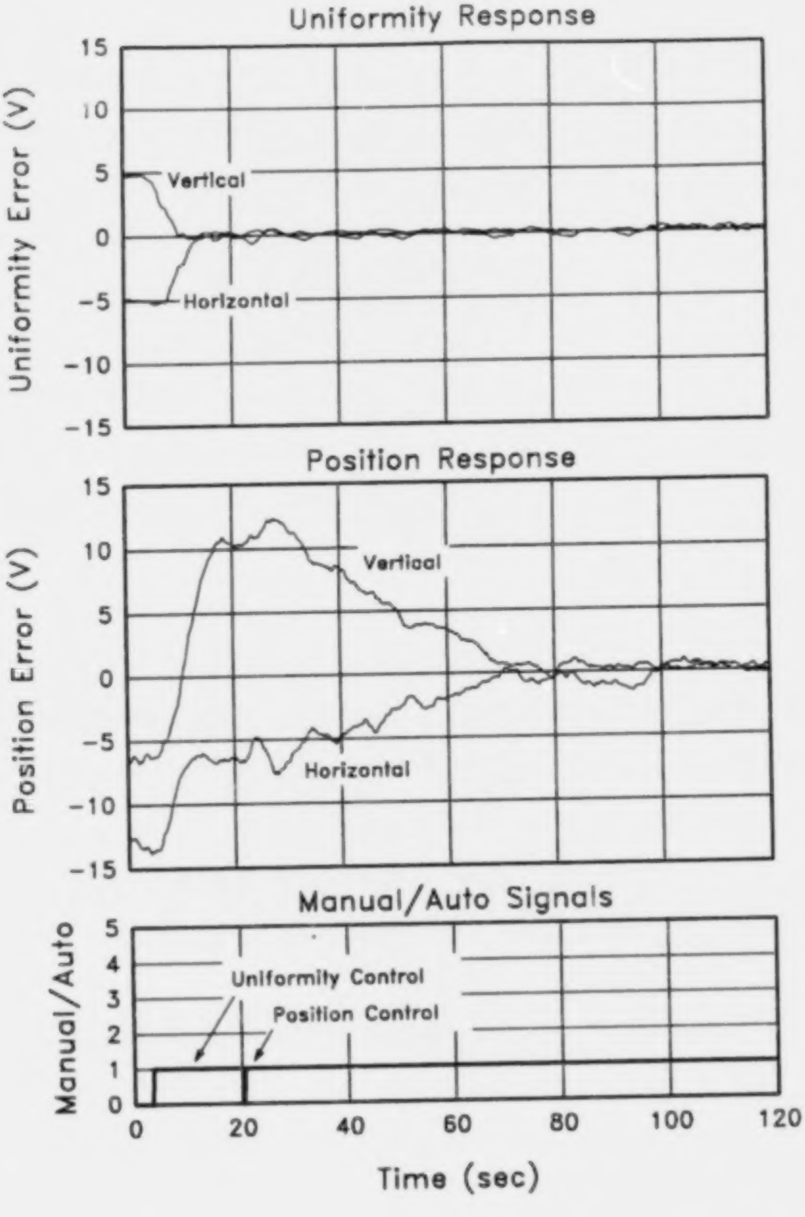


Fig. 6.15 Performance of the Alignment Controller in Recovering from a Severe Misalignment of both Resonator Mirrors

6.6.3 Steady-State Performance

The ability of the mode controller to maintain mode quality for extended periods was tested during a 30 minute run. The laser was run from a cold start for a 30 minute period with the power, uniformity and position controllers on automatic. At approximately the 24 minute mark in the run, a small power level disturbance was introduced by manually adjusting the discharge current setpoint. Then, at about 27 minutes, the power level setpoint was changed from 3.5 kW to 5.0 kW. Laser output power, and the uniformity and position error signals are plotted in Fig. 6.16.

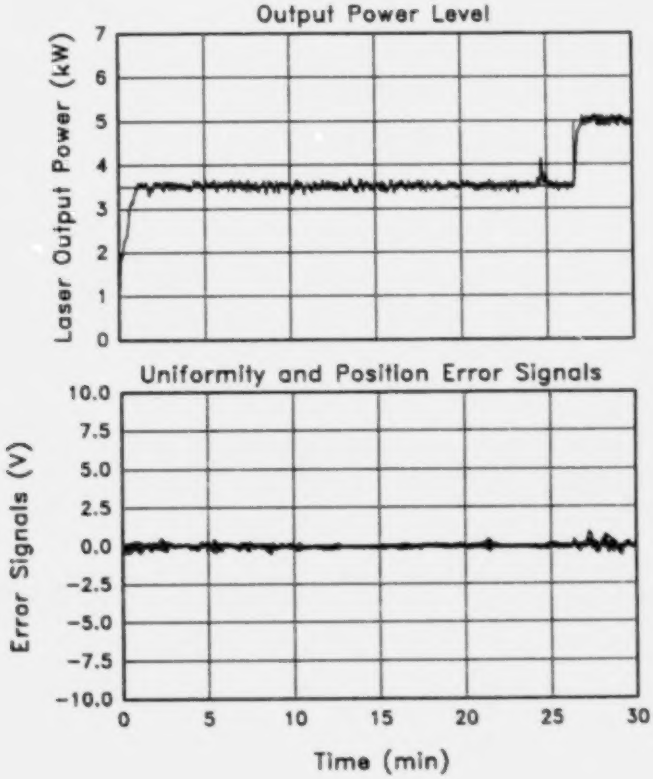


Fig. 6.16 Steady-State Performance of the PIE-4 Mode Controller

6.7 Discussion

Fig. 6.14 reveals a critically damped response from the power control portion of the system. A step change in power level is performed by the controller within approximately 15 seconds. This performance could be considerably improved by equipping the laser with a second rotating wand and a smaller thermopile type power meter, as was done in the experiments with the PIE-3 laser. Alternatively, since an accurate model of the power system is available, one could employ a Smith predictor controller, with the measured discharge current as the input to the computer model. Using this method, step changes in power could be performed very quickly. Depending on the process requirements this technique could be quite useful.

As is evident in Fig. 6.15, the addition of feed-forward compensation to the alignment controller considerably reduced the disturbances to mode uniformity previously caused by the position controller. It can be seen from the uniformity response plot that the system restores a uniform mode within approximately 5 seconds, and that uniformity remains constant as the position portion of the system recovers. It should be noted that during the recovery time of the uniformity signals, the position error signals also undergo a rapid change. This is as expected since the position signals do not provide an accurate indication of beam location until a uniform mode is present. The position response plot reveals that the alignment system required approximately 50 seconds to recover from the relatively large (50 μm) misalignment used in this test. It was observed that under normal operating conditions the maximum misalignment would be much less.

Fig. 6.16 demonstrates clearly the controller's ability to maintain mode quality and output power level for extended periods. As shown in the plot, power level is maintained precisely at the desired setpoint, and disturbances to power level are quickly compensated. It is apparent from the alignment signal plot that changes in the power setpoint disturb mode uniformity and position. The controller effectively compensates during these setpoint changes to maintain mode quality.

6.8 Summary

The primary goal of the experiments described in this chapter, was to provide an effective demonstration that automatic mode control is a viable design technique in industrial lasers. Therefore, a more permanent installation was designed for the 25 kW PIE-4 laser.

The PIE-4 laser's optical system was modified to allow accurate study of the open-loop alignment system. This conversion was accomplished by simply replacing the existing motormics with digital readout versions. The power supply system of the laser was also modified slightly, to allow feedback stabilization of the laser's output intensity. Here, the only requirement was the addition of an auxiliary setpoint input to the discharge current controller.

The mode controller's circuitry was placed on printed circuit boards, and mounted in an electronics rack already attached to the laser enclosure. The sensor array and rotating wand were then mounted on the laser, and arrangements were made to interface the data-acquisition system with the mode controller electronics.

At this stage, several enhancements were made to the control/data-acquisition system software. A graphical user interface was designed which permits the operator to select from a number of data sources, and perform on-line tuning of the power and alignment controllers. Digital filtering was incorporated into this system to enhance the diagnostic capabilities of the system. The computer was also utilized to implement a refined alignment and power control system. Power control of the laser was performed with a standard, digital, PI controller. Alignment controller performance was improved by introducing a form of feed-forward compensation into the position control system. Experimental tests demonstrated that this method of position control significantly reduced disturbances to beam uniformity. Mode uniformity was controlled using the convex mirror, as in previous experiments.

Finally, experimental data were collected to document the transient and steady-state performance of the combined alignment and power control systems. It was found that from a cold start, complete resonator alignment could be obtained within a maximum of approximately 1 minute. Observations revealed that under normal conditions this period was reduced to about 15 seconds, or the time required for the power controller to stabilize. Although most of the experimental data was collected below the 5 kW level, simply for convenience, the controller has been tested at power outputs of up to 10 kW. Beyond this level, the pyroelectric detectors begin to saturate, and steps must be taken to reduce the intensity of the sample provided to the sensor array. It was found that the addition of a second metal screen, between the rotating wand and the sensor array, performed this function adequately. In lasers with large dynamic ranges, arrangements to automatically insert a second screen could be made, if operation over the full range of output power was required. In many applications, operation is confined to within 1 or 2 kW of a chosen power level. In such cases the system could be easily configured to perform properly over the desired range.

CHAPTER 7

SUMMARY AND RECOMMENDATIONS

The successful testing of the automatic mode controller on the 25 kW PIE-4 laser brings the project to a logical close. Although by no means perfected, the system has clearly demonstrated that improved performance, and overall reduced laser cost, can be achieved using a properly designed feedback controller. At this point it is appropriate to summarize the project, and to provide some recommendations as to further development of the system.

7.1 Summary

In the early stages of the work, an array of thermistors was used as a beam profile sensor. A rotating wand beam sampler was used to couple a small portion of the laser's output onto this array and a feedback control circuit was developed to drive a pair of motor-micrometers, mounted on the convex resonator mirror, so as to continuously maintain beam uniformity. Although this system clearly demonstrated the feasibility of an automatic resonator alignment system, the feedback sensor's response characteristics were less than ideal. The long time constant of the thermistor elements restricted the controller usefulness to applications requiring only long term stabilization of mode uniformity. This led to the development of a new sensor, substituting pyroelectric detectors for the thermistors.

Experiments with this new array demonstrated significantly superior response characteristics. Although the new array employed slightly more complicated interface circuitry, this requirement was a minor inconvenience in return for its markedly improved performance. In addition to faster response, the new array supplied a means for controlling the second primary mirror, thus allowing completely automatic resonator alignment. This was accomplished by configuring the sensor array to detect shifts in the position of the annular beam sample.

During the experiments with the pyroelectric sensor array, a microcomputer based data-acquisition system was installed, to facilitate accurate theoretical study of the system. With the aid of this computer, a model of the power, uniformity, and position control loops was constructed. Simulations of this model were subsequently employed to study the performance of various controller structures, and to obtain initial tuning parameters for the closed loop system. Eventually, an electronic, non-linear control system was constructed for the 5 kW PIE-3 laser. Experiments with this controller demonstrated that beam position could be stabilized by adjusting the concave mirror while continuously aligning the convex mirror so as to maintain beam uniformity. Power output stabilization of this laser was accomplished by using a relay controller to activate the laser's variac power supply.

To complete the project, a prototype controller, suitable for installation on a 25 kW PIE laser, was constructed. In addition to providing a challenging test for the system, this modern laser provided an opportunity to test more advanced control strategies for the power and position control loops. The computer based data-acquisition system was transferred to the new laser, and used to implement these new control strategies. A number of enhancements were also made to the data-acquisition software, in order to improve its usefulness as a diagnostic tool. Among the more important of these was the addition of a graphical user interface. This utility permits the user to view data from various sources, and to perform on-line modification of controller tuning parameters.

In the power control loop, a digital PI controller was used to replace the relay controller used on the PIE-3 laser. Unfortunately, the rapid response capability of the PIE-4 laser's SCR power supply was not fully utilized because of the power meter's slow response. It was thought that improvement in this area could be obtained using more advanced controller structures, such as the Smith predictor. The experiments successfully demonstrated that a classical PI controller structure could be employed to obtain accurate, steady-state power regulation on lasers equipped with SCR power supplies.

During experiments with the PIE-3 laser, beam position movements were performed by moving the concave mirror; the convex mirror was then independently adjusted as necessary to maintain mode uniformity. This technique, although it provides adequate performance, creates large uniformity disturbances, particularly if the speed of the concave mirror micrometers is not kept at relatively low levels. To overcome this problem, a controller which performs position movements by simultaneously activating the servos on both resonator mirrors, was developed. Beam uniformity is maintained using only the convex mirror servos, as before. Experimental results revealed a substantial reduction in uniformity disturbances, when using this method.

Although the controller is able to accomplish complete resonator alignment, it was seen that the response time of the position controller was still too slow to provide good performance in systems requiring frequent changes in power level. This was not attributed to a problem with servos, or the feedback sensor, but rather, to the method by which changes in beam position are detected. In order to obtain error signals which describe the beam position, it is necessary to maintain the mode structure as close to uniform as possible. Because of the high frequency optical noise generated by the laser system, it is necessary to low-pass filter the error measure signals. The position error measures are filtered at a much lower frequency than the uniformity signals. This was done to prevent short term fluctuations in beam uniformity from influencing the position signals. When the power level is altered suddenly, there are unavoidable disturbances to the laser's mode uniformity and position. The uniformity controller settles quickly, but the position portion of the system responds more slowly. If changes in power level are made frequently, the position controller begins to hunt constantly as it attempts to properly position the output beam. Although the feed-forward compensation scheme used in the PIE-4 alignment controller alleviated this problem somewhat, it was still necessary to make position adjustments at a relatively slow rate. In material processing applications which require frequent changes in output power level, it would seem that a suitable compromise would be to forego the position controller, and simply operate only the uniformity control portion of the system. It was observed that in many situations this method would yield better laser performance than is provided by the open-loop systems currently employed in many industrial installations.

7.2 Recommendations for Future Research

At this point a few recommendations may be made as to how to improve the performance of the system, and to make it more amenable to transfer to industry. The first improvement which could be made involves the alignment servos. It does not seem possible to obtain an inexpensive motor-micrometer with high torque, slow speed characteristics that is suitable for installation in the demanding environment encountered inside multikilowatt gas lasers. During the project, motor-micrometers were used which provided digital feedback of the shaft position, and closed loop stabilization of the micrometer's rotation speed. Although these units allowed completion of the project, they exhibited two main problems. They provided only marginally sufficient torque, and were susceptible to interference from nearby electrical noise sources. Because of their low torque characteristics, a non-linear system was found to be the only viable controller structure. The development of heavy duty motor-micrometers, capable of continuously variable operation at very low rotation speeds, would permit the application of conventional linear system control techniques.

The second component which could be improved upon, is the pyroelectric sensor array. It was found that the spacing between array elements rendered the array ineffective as a feedback element for controlling the laser's output power. It is felt that the development of a circular array, composed of pyroelectric crystals mounted in close proximity, would solve this problem. Such an array would provide a fast, accurate indication of total laser power while simultaneously eliminating the need for a separate power meter. It would also improve the performance of the alignment controller, by providing improved spatial resolution. Although there are currently a few rectangular arrays of this type on the market, they are very small in size, and prohibitively expensive. Perhaps, as has occurred in other areas of electronics, the price of these units will drop sufficiently to allow their incorporation into automatic alignment systems. It would also be worthwhile to conduct an investigation into the suitability of other types of sensors for constructing an array of this type.

In conclusion, the automatic mode controller has quite successfully demonstrated its ability to provide significant improvements in high power laser mode quality and consistency, as compared to those systems currently operating in an open-loop mode. Even at its present level of sophistication, the system could be incorporated into industrial laser systems to improve performance and reduce cost.

REFERENCES

1. C. Patel, Phys. Rev. Lett. Vol. 336 A., pp. 1187 (Nov, 1964)
2. J. E. Harry, Ed., Industrial Lasers and Applications (McGraw Hill: London 1974)
3. J. Ready, Ed., Lasers in Modern Industry. Soc. Manufact. Eng. (Dearborn, MI, 1979) pp. 10-11
4. J. S. Eckersley, "**Laser applications in metal surface hardening**," Proc. SME Conf. "Laser Welding and surface treatment," Plymouth, MI, (Oct. 1985)
5. J. E. Harry, Industrial Lasers and Applications, (McGraw Hill, London, 1974)
6. I.J. Spalding, A.C. Selden, J.H.P.C. Megaw, and B.A. Ward, "**High power CO2 lasers**," Proc. 7th Int. Symp. Gas Flow and Chemical Lasers, Vienna, Austria (Aug. 1988)
7. Y. Arata, Plasma, Electron & Laser Beam Technology. (Amer. Soc. Metals: Ohio, 1986)
8. A. Siegman, Lasers, (University Science Books: Mill Valley, CA, 1986)
9. V.E. Merchant, M.R. Cervenan, H.J.J. Seguin, "**An Industrial Quality 20 kW. Infrared Laser**", Proc. Int. Conf. Lasers (1985)
10. A.J.B. Travis, "**Laser Beam Diagnostic Equipment for On-lin Use with Multikilowatt CO2 Lasers**", Proc. 5th Int. Symp. Gas Flow and Chemical Lasers, pp. 367-372 (1984)
11. J.T. Knudtson, K.L. Ratzlaff, "**Laser beam spatial profile analysis using a two-dimensional photodiode array**", Rev. Sci. Inst., Vol. 54, pp. 856-860 (1983)
12. H.G. Heard, Laser Parameter Measurements Handbook, (John Wiley & Sons, Inc.: New York, NY, 1968)
13. P.D. Austin, "**High Power CO2 Laser Beam Diagnostics**", Proc. SPIE-Laser Processing: Fundamentals, Applications and Controls, pp. 232-235, (1986)
14. G.C. Lim and W.M. Steen, "**Laser Beam Analyzer**", Proc. 1st Int. Conf. on Lasers in Manufact., Brighton, U. K., pp. 161-167, (1983)

15. G. Sepold, P.O. Juptner, J. Telepski, "**Measuring the Quality of High Power Laser Beams**", Proc. SPIE-High Power Lasers and their Industrial Applications, D. Schuocker ed., pp 167-169, (1986)

16. J.E. Harvey and M.L. Scott, "**Hole Grating Beam Sampler-Versatile High-energy Laser Diagnostic Tool**", Opt. Eng. Vol. 20, pp. 881-886, (1981)

17. T.J. Ramos, D.R. Lim, A.C. Lingenfelter, "**Low-cost Laser Diagnostic System**", Proc. Medicine and and Biology Symp., ICALEO, pp. 152-157, (1985)

18. F.H. White, G.A. Needham, "**High Energy Laser Diagnostics-A Review**", Proc. SPIE-Laser Diagnostics, pp. 2-15, (1982)

19. F. Martin, J.G. Willman, "**Video Graphics System Analyzes Laser Beam Profiles**", Laser Focus/Electro. Opt. (USA), Vol. 21, 104-111, (1985)

20. A. Siegman, **Lasers**, University Science Books: Mill Valley, CA (1986)

21. Instruction Manual, Laakman model RF-125 CO2 laser

22. Operating manuals, ORIEL model 18009 motor-micrometer controller, model 18212-18279 motor-micrometers, Oriel Corp., Stratford, CN, 06497-0872.

23. Melles Griot Optics Guide, pp. 196-197, (1985)

24. Molectron PQD-220 Pyroelectric Quadrant Detector Manufacturers data sheets

25. C.V. Sellathamby, H.J.J. Seguin, S.K. Nikumb, "Performance characteristics of a high power CO2 laser with computer vision mode and power control", Appl. Opt., Vol. 29, No. 30, pp. 4499 (Oct. 1990)

26. A.K. Nath, H.J.J. Seguin, V.A. Seguin, "Optimization studies of a multikilowatt PIE CO2 laser", IEEE Jour. Quantum Electron., Vol. QE-22, pp. 268-274 (1986)

27. S.K. Nikumb, H.J.J. Seguin, V.A. Seguin, H. Reshef, "**Gain and saturation parameters of a multikilowatt PIE CO2 laser**", J. Phys. E., Vol. 20, pp. 911 (1987)

28. Z. Cheng, H.J.J. Seguin, S.K. Nikumb, V.A. Seguin, H. Reshef, "**Annular-coupled concave-convex stable resonator for large-volume high-quality energy extraction**", Appl. Opt., Vol. 27, pp. 836 (Mar. 1988)

29. W.G. Jung, **IC Op-Amp Cookbook**, (H. W. Sams: Indianapolis, IN, 1978)

30. W.T. Baker, "**The Thermopile: it is and it isn't**", Optical Spectra (Mar/1977)

31. Thin Film Thermopile Detectors Specification Sheet, Sensors Inc.

32. Introduction to Infrared Pyroelectric Detectors, Eltec Instruments, Inc. (Application note #100)

33. H. Keller, D. Cima, "**Choosing and using Pyroelectric Detectors**", Eltec Instruments, Inc. (Application note #101)

34. W.G. Jung, **IC Op-Amp Cookbook**, (H. W. Sams: Indianapolis, 1978)

35. Model 203 Power Meter Instruction Manual, Coherent Radiation Labs, Palo Alto, CA

36. J.R. Leigh, **Essentials of nonlinear control theory**, (P. Peregrinus Ltd.: London, 1983)

37. E.P. Ryan, **Optimal relay and saturating control system analysis**, (P. Peregrinus Ltd.: London, 1982)

38. D.S. Dragoslav, **Nonlinear Systems-The parameter analysis and design**, (J. Wiley and Sons: New York, 1969)

39. D. Akitt, H.J.J. Seguin, M.R. Cervenan, and S.K. Nikumb, "**Automatic Alignment of a High-Power CO2 Laser Resonator**", Appl. Phys. B, Vol. 51 pp. 326-328 (1990)

40. D.R. Akitt, H.J.J. Seguin, M.R. Cervenan, S.K. Nikumb, "**Electronic Mode and Power Control of a High-Power CO2 Laser**", IEEE J. Quantum Electronics, Vol. 26, no. 8 pp. 1413 (1990)

41. TUTSIM users manual, Applied i, 200 California Ave, #212, Palo Alto, CA 94306

42. L. Ljung, T. Soderstrom, **System Identification: Theory for the User**, (Prentice-Hall, 1987)

43. J.R. Leigh, **Essentials of nonlinear control theory**, (P. Peregrinus Ltd.: London, 1983)

44. E.P. Ryan, **Optimal relay and saturating control system analysis**, (P. Peregrinus Ltd.: London, 1982)

45. D.S. Dragoslav, **Nonlinear Systems-The parameter analysis and design**, (J. Wiley and Sons: New York, 1969)

46. H. Hostetter, J.S. Clement, R.T. Stefani, **Design of Feedback Control Systems**, (Holt, Rinehart and Winstor: New York, 1982)

47. D.P. Atherton, **Stability of Nonlinear Systems**, (J. Wiley & Sons Ltd.: New York, 1981)

48. J.R. Leigh, Essentials of nonlinear control theory, (P. Peregrinus Ltd.: London, 1983)

49. E.P. Ryan, Optimal relay and saturating control system analysis, (P. Peregrinus Ltd.: London, 1982)

50. R.A. Heyler, S.C. Guggenheimer, "**Stabilized Beam Position Improves Ion-Laser System Performance**", Ion-Laser Technology.

51. A. Siegman, Lasers, (University Science Books: Mill Valley, CA 1986)

52. W.F. Krupke, W.R. Sooy, "**Properties of an Unstable Confocal Resonator CO2 Laser System**", IEEE Journal of Quantum Electronics, Vol. QE-5 No. 12 (Dec 1969)

53. C.V. Sellathamby, H.J.J. Seguin, S.K. Nikumb, "**Performance characteristics of a high power CO2 laser with computer vision mode and power control**", Appl. Opt., Vol. 29, No. 30, pp. 4499 (Oct. 1990)

54. V.E. Merchant, "**Development of a New 20-kW CO2 Laser**", Laser Focus/Electro-Optics, (May 1985)

55. Operating manuals, ORIEL model 18009 motor-micrometer controller, model 18212-18279 motor-micrometers, Oriel Corp., Stratford, CN, 06497-0872

56. K.J. Astrom and B. Wittenmark, Computer Controlled Systems: Theory and Design, (Prentice Hall, 1984)

APPENDIX 1 - DIGITAL FILTER DESIGN

1. First Order Lowpass Filter

A first order, low-pass digital filter was obtained by applying the bilinear transform to a low-pass analog filter as shown below:

$$\mathbf{H}(s) = \frac{1}{s\tau + 1} = \frac{1}{(s/\omega_c) + 1}$$

$$\mathbf{H}(z) = \mathbf{H}(s)\Bigg|_{\left[s = \frac{2(1-z^{-1})}{T_s(1+z^{-1})}\right]} = \frac{a_0 + a_1 z^{-1}}{b_0 + b_1 z^{-1}}$$

where:

$$a_0 = 1 \quad b_0 = 1 + \tfrac{2}{\omega_c T_s}$$

$$a_1 = 1 \quad b_1 = 1 - \tfrac{2}{\omega_c T_s}$$

T_s = sampling period (sec)

ω_c = filter cutoff frequency (rads / sec)

The corresponding difference equation is:

$$y(n) = (\tfrac{a_0}{b_0})x(n) \;+\; (\tfrac{a_1}{b_0})x(n-1) \;-\; (\tfrac{b_1}{b_0})y(n-1)$$

2. Third order Butterworth filter

A filter with improved rolloff characteristics was obtained by turning to the Butterworth filter. Frequency scaling was first applied to a normalized, 3rd-order Butterworth low-pass filter. Subsequent application of the bilinear transform provided the corresponding digital filter. The procedure is as follows:

$$H(s) = \frac{1}{1+2s+2s^2+s^3}\Bigg|_{\left[s=\frac{s}{\omega_c}\right]} = \frac{1}{1+2(\frac{s}{\omega_c})+2(\frac{s}{\omega_c})^2+(\frac{s}{\omega_c})^3}$$

$$H(z) = H(s)\Bigg|_{\left[s=\frac{2(1-z^{-1})}{T_s(1+z^{-1})}\right]} = \frac{a_0+a_1z^{-1}+a_2z^{-2}+a_3z^{-3}}{b_0+b_1z^{-1}+b_2z^{-2}+b_3z^{-3}}$$

where:

$$a_0 = (\omega_c)^3 \quad b_0 = \frac{8}{(T_s)^3} + \frac{8\omega_c}{(T_s)^2} + \frac{4(\omega_c)^2}{T_s} + (\omega_c)^3$$

$$a_1 = 3(\omega_c)^3 \quad b_1 = \frac{-24}{(T_S)^3} - \frac{8\omega_c}{(T_s)^2} + \frac{4(\omega_c)^2}{T_s} + 3(\omega_c)^3$$

$$a_2 = 3(\omega_c)^3 \quad b_2 = \frac{24}{(T_S)^3} - \frac{8\omega_c}{(T_s)^2} - \frac{4(\omega_c)^2}{T_s} + 3(\omega_c)^3$$

$$a_3 = (\omega_c)^3 \quad b_3 = \frac{-8}{(T_s)^3} + \frac{8\omega_c}{(T_s)^2} - \frac{4(\omega_c)^2}{T_s} + (\omega_c)^3$$

T_s = sampling period (sec)
ω_c = filter cutoff frequency (rads / sec)

The corresponding difference equation is:

$$\begin{aligned} y(n) = {} & (\tfrac{a_0}{b_0})x(n) + (\tfrac{a_1}{b_0})x(n-1) + (\tfrac{a_2}{b_0})x(n-2) + (\tfrac{a_3}{b_0})x(n-3) \\ & - (\tfrac{b_1}{b_0})y(n-1) - (\tfrac{b_2}{b_0})y(n-2) - (\tfrac{b_3}{b_0})y(n-3) \end{aligned}$$

APPENDIX II - TUTSIM SIMULATIONS

This appendix contains details of the procedures and model structures employed to simulate the closed-loop PIE-3 system. Simulations were performed using TUTSIM (Twente University of Technology Simulation Program). This program allows simulation of both linear, and non-linear blocks in a fashion similar to the use of an analog computer, but with added versatility and accuracy.

The block diagrams are readily related to the text based "programming language" used by the TUTSIM simulation program. In some cases more complicated blocks are defined as macros. For more detail, the interested reader is referred to the TUTSIM user's manual.

1. Alignment System Simulation

The lack of interaction between the vertical and horizontal alignment axes allowed simulation of the alignment system using a single program. Tuning of each axis was then performed by modifying the appropriate parameters to accurately reflect the axis under consideration.

Fig. 1 illustrates the TUTSIM simulation utilized to study each alignment axis. Each block is named and numbered to correspond with the text listing of the model structure as entered inside TUTSIM. Fig. 2 provides details for the controller macro which is the central component of the simulation. Two other macros used to implement the controller macro, a resettable pulse generator and a relay with a dead-zone, are illustrated in Figs. 3 and 4. Following Fig. 4, a complete listing of the model structure is provided in TUTSIM format.

2. Power System Simulation

The simulation diagram for the power control loop appears in Fig. 5. The controller macro used in this loop is illustrated in Fig. 6. The pulse generator and relay macros, also used in the power controller, are shown in Figs. 3 and 4. A complete listing of this simulation is provided following Fig. 6.

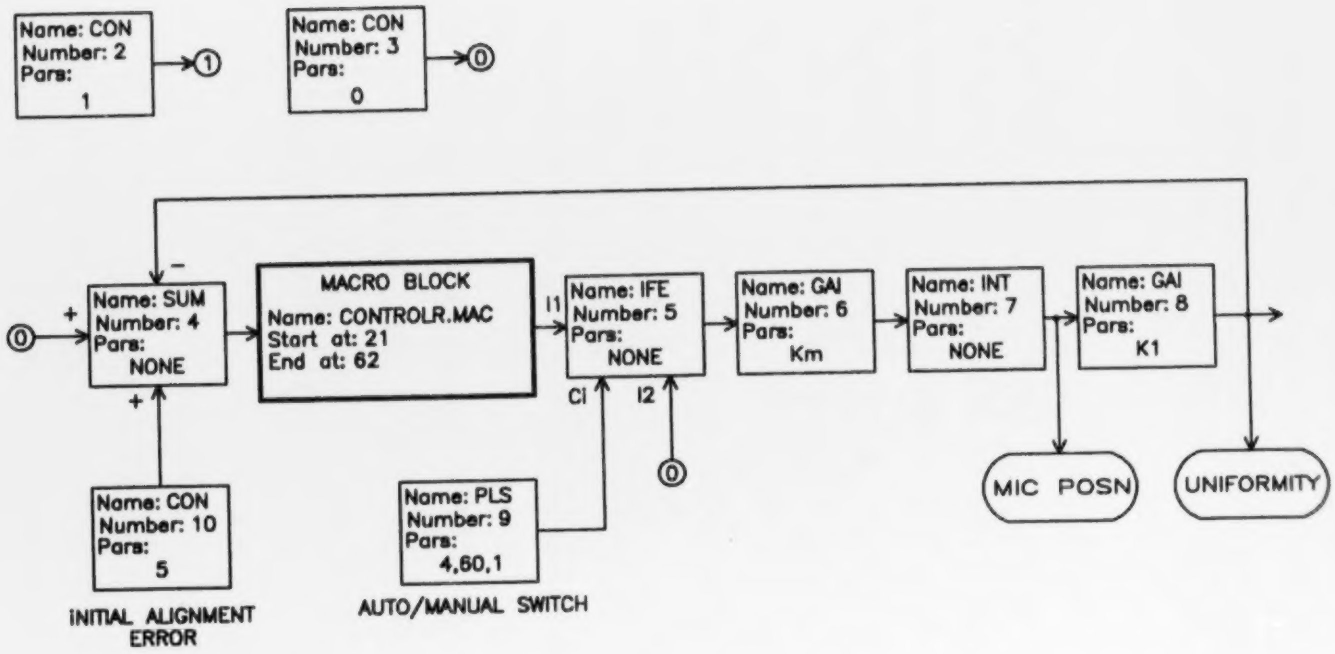


Fig. 1 Closed-Loop Alignment System Simulation Diagram (single axis)

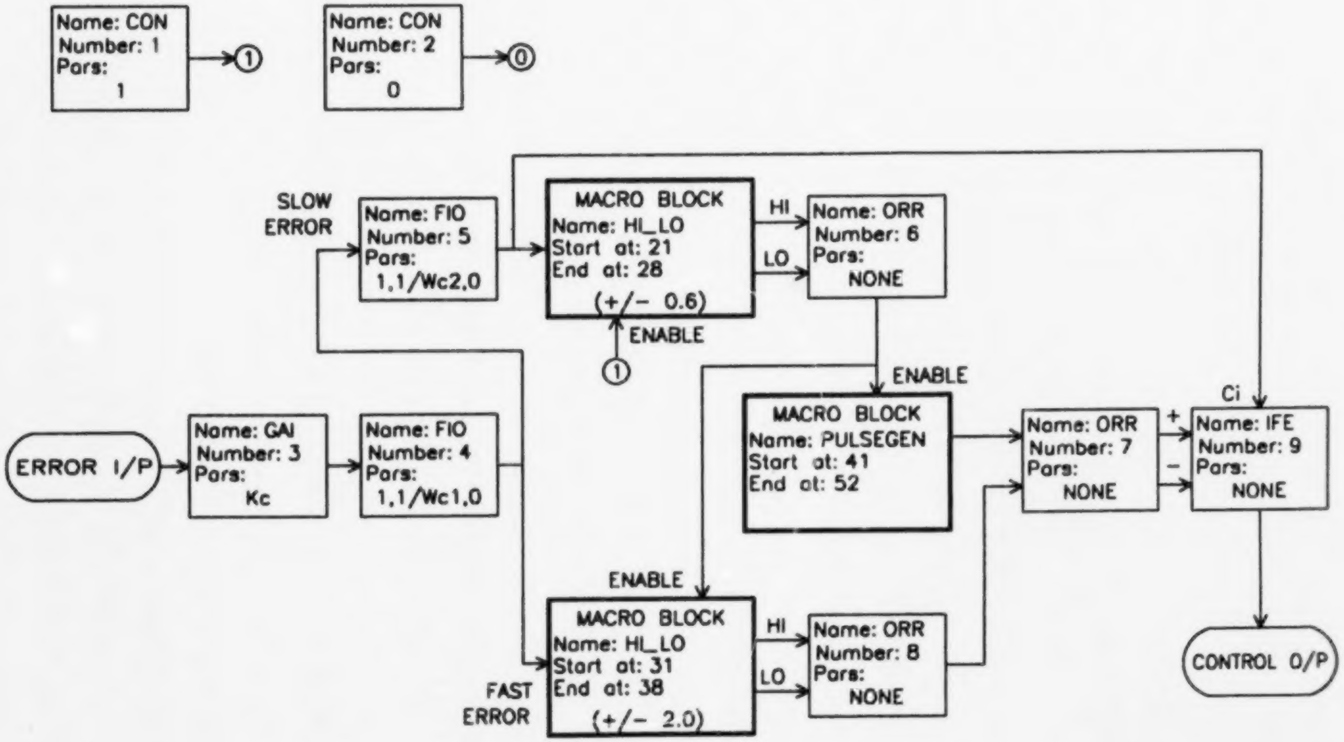


Fig. 2 Non-linear Alignment Controller Macro

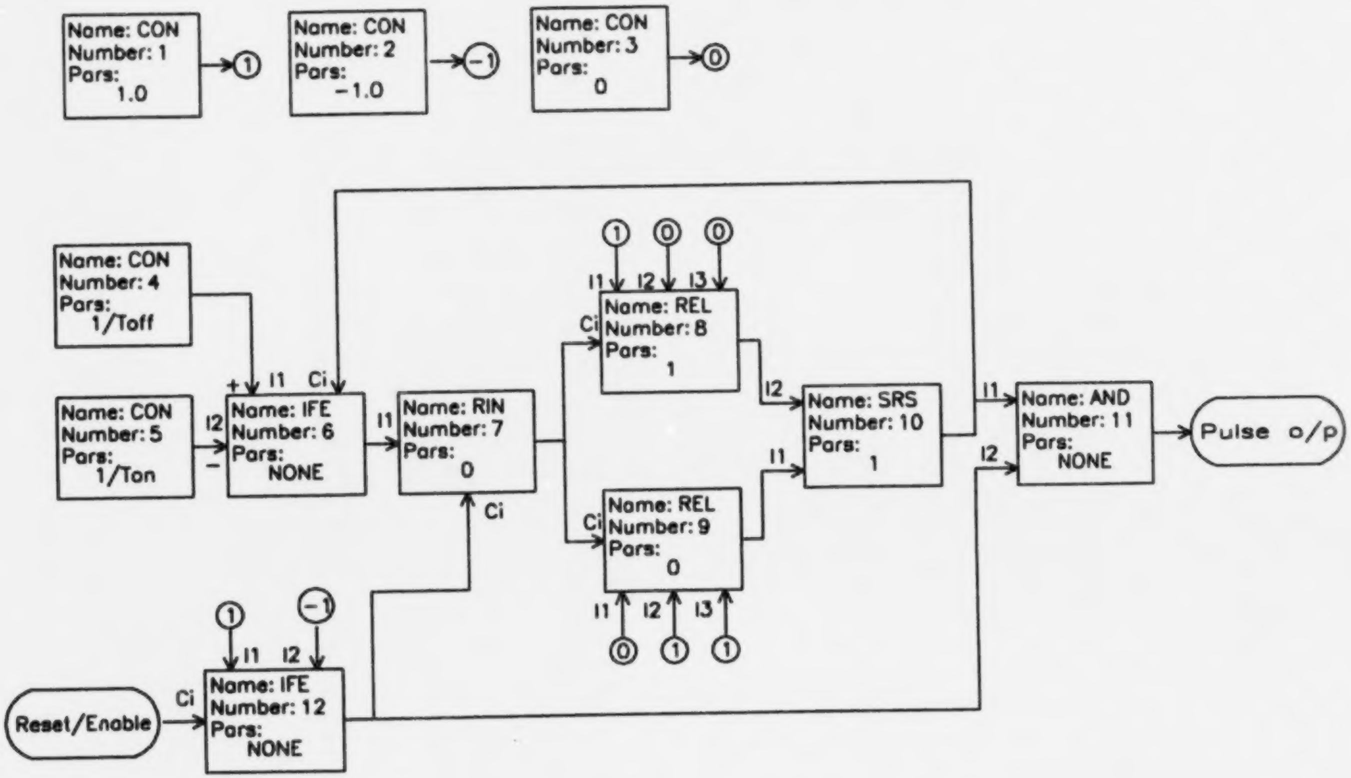


Fig. 3 Resettable Pulse Generator Macro

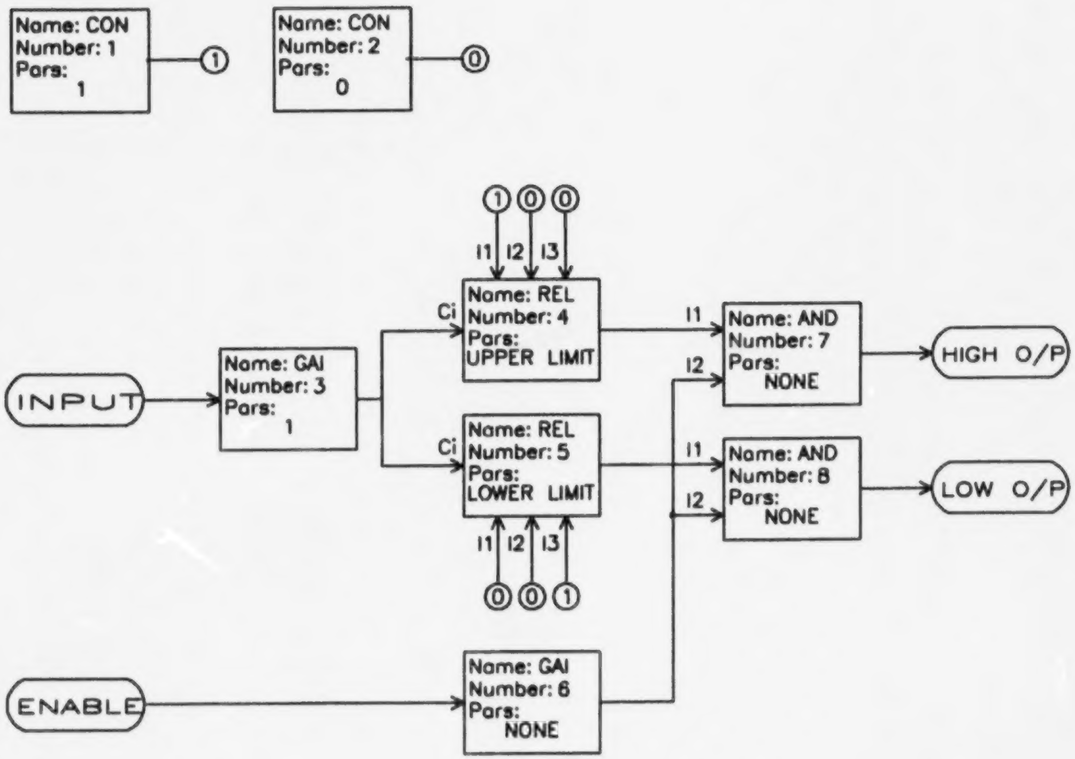


Fig. 4 Relay with Dead-Zone Macro

3. Alignment Simulation Model Listing

```
Model File: D:\TUTSIM\PIE3\VERTSIM.MOD
Date:     2 /    28 /  1992
Time:    12 :     0
Timing:   0.0100000  ,DELTA  ;  38.0000     ,RANGE
PlotBlocks and Scales:
Format:
     BlockNo,  Plot-MINimum,   Plot-MAXimum;  Comment
Horz:      0 ,   0.0000     ,  38.0000     ; Time
  Y1:     11 , -10.0000     ,  10.0000     ; MEASURED UNIFORMITY ERROR
  Y2:     25 , -10.0000     ,  10.0000     ; SIMULATED UNIFORMITY ERROR
  Y3:      9 , -10.0000     ,  10.0000     ; AUTO/MANUAL CONTROL
  Y4:        ,              ,              ;


                      1 REM
;BLOCKS 2-10 SIMULATE THE OPEN-LOOP SYSTEM
   1.0000             2 CON                         ;1 SOURCE
   0.0000             3 CON                         ;0 SOURCE
                      4 SUM      3      10      -8  ;INPUT SUMMER
                      5 IFE      9      29       3  ;CONTROL SIGNAL
  10.0000             6 GAI      5                  ;MICROMETER SPEED (u/sec)
   0.0000             7 INT      6                  ;MICROMETER POSITION
   0.0780000          8 GAI      7                  ;AXIS SENSITIVITY (V/u)
   6.0000             9 PLS                         ;AUTO/MANUAL CONTROL
  38.0000
   1.0000                                           ;INITIAL ERROR
  -1.6000            10 CON
   2.0000            11 DAT                         ;MEASURED UNIFORMITY ERROR
                     12 REM
;CLOSED-LOOP CONTROLLER BEGINS HERE                 ;LOGIC ONE SOURCE
   1.0000            21 CON                         ;LOGIC ZERO SOURCE
   0.0000            22 CON                         ;CONTROLLER GAIN (Kc)
   4.0000            23 GAI      4                  ;FAST FILTER
   1.0000            24 FIO     23
   0.2040000
  -6.4000
   1.0000            25 FIO     24                  ;SIMULATED UNIFORMITY ERROR
   0.9950000
  -6.4000
                     26 ORR     47      48
                     27 ORR     71      28
                     28 ORR     57      58
                     29 IFE     25      27      31  ;CONTROLLER OUTPUT
  -1.0000            31 GAI     27
                     40 REM
;RELAY WITH DEAD-ZONE (SLOW CONTROLLER)             ;LOGIC ONE SOURCE
   1.0000            41 CON             .           ;LOGIC ZERO SOURCE
   0.0000            42 CON                         ;SIGNAL INPUT (SLOW)
   1.0000            43 GAI     25
   0.6000000         44 REL     41      42      42  ;HIGH RELAY
                                43
  -0.6000000         45 REL     42      42      41  ;LOW RELAY
                                43
   1.0000            46 GAI     21                  ;ENABLE INPUT
                     47 AND     44      46          ;HIGH O/P
                     48 AND     45      46          ;LOW O/P
                     50 REM
;RELAY WITH DEAD-ZONE (FAST CONTROLLER)             ;LOGIC ONE SOURCE
   1.0000            51 CON                         ;LOGIC ZERO SOURCE
   0.0000            52 CON                         ;SIGNAL INPUT (FAST)
   1.0000            53 GAI     24
   2.0000            54 REL     51      52      52  ;HIGH RELAY
                                53
  -2.0000            55 REL     52      52      51  ;LOW RELAY
                                53
   1.0000            56 GAI     26
                     57 AND     54      56          ;HIGH O/P
                     58 AND     55      56          ;LOW O/P
                     60 REM
```

```
;RESETTABLE PULSE GENERATOR
   1.0000         61 CON                       ;LOGIC ONE SOURCE
  -1.0000         62 CON                       ;-1.0 SOURCE
   0.0000         63 CON                       ;LOGIC ZERO SOURCE
  20.0000         64 CON                       ;1/Toff
   2.0000         65 CON                       ;1/Ton
                  66 IFE     70     64   -65
   0.0000         67 RIN     72     66
   1.0000         68 REL     61     63     63
                             67
   0.0000         69 REL     63     61     61
                             67
   1.0000         70 SRS     69     68
                  71 AND     70     72         ;PULSE OUTPUT
                  72 IFE     26     61     62  ;RESET/ENABLE INPUT

END OF MODEL
```

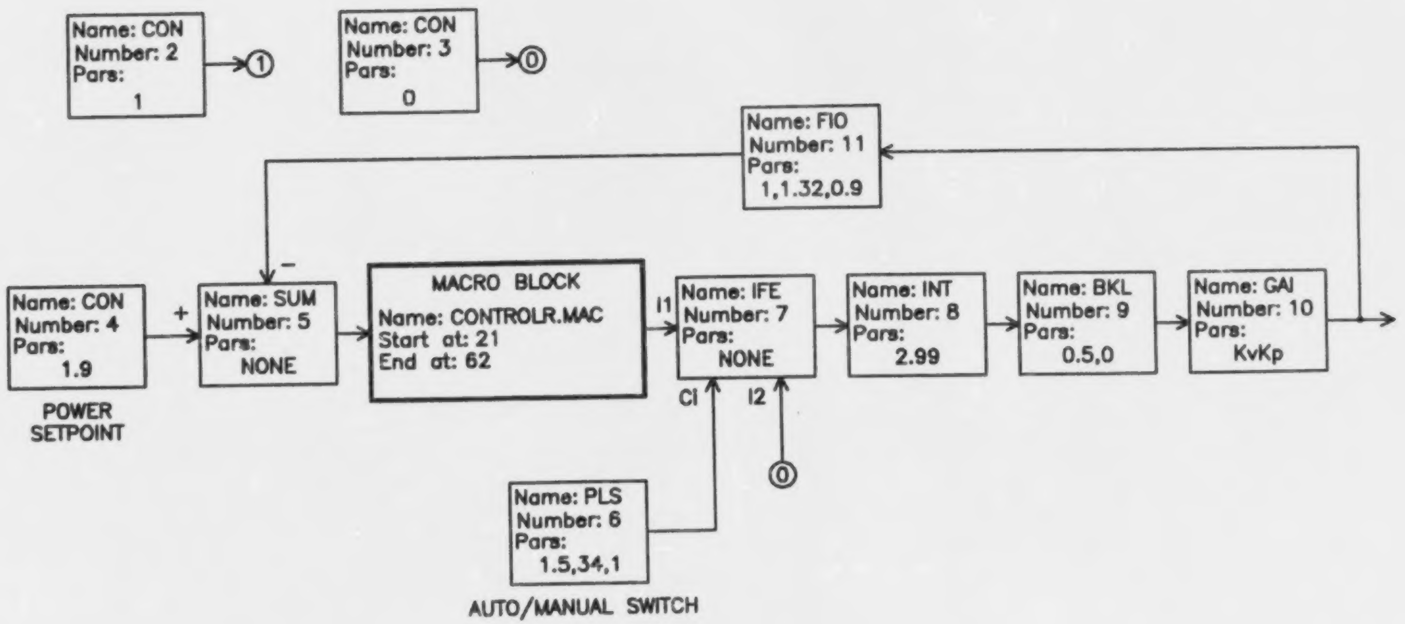


Fig. 5 Power Control Loop Simulation Diagram

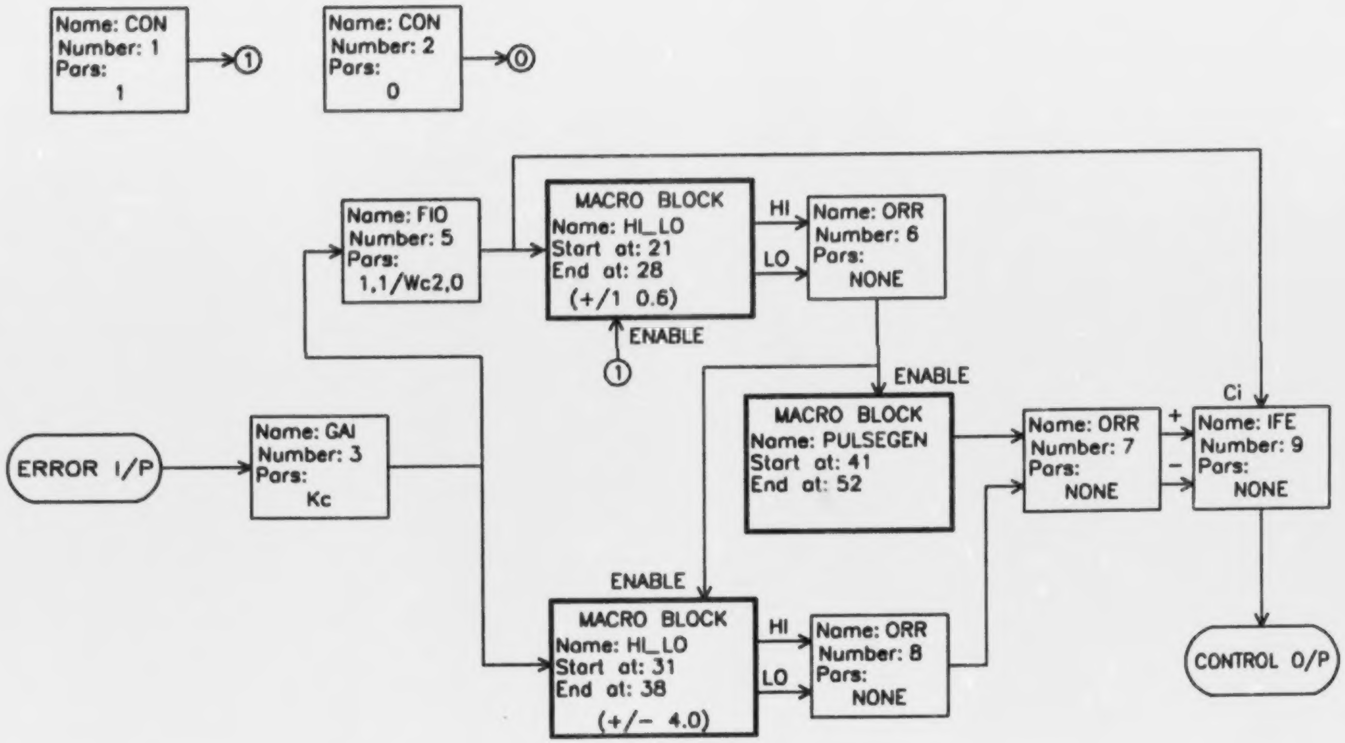


Fig. 6 Non-linear Power Controller Macro

4. Power Control Simulation Model Listing

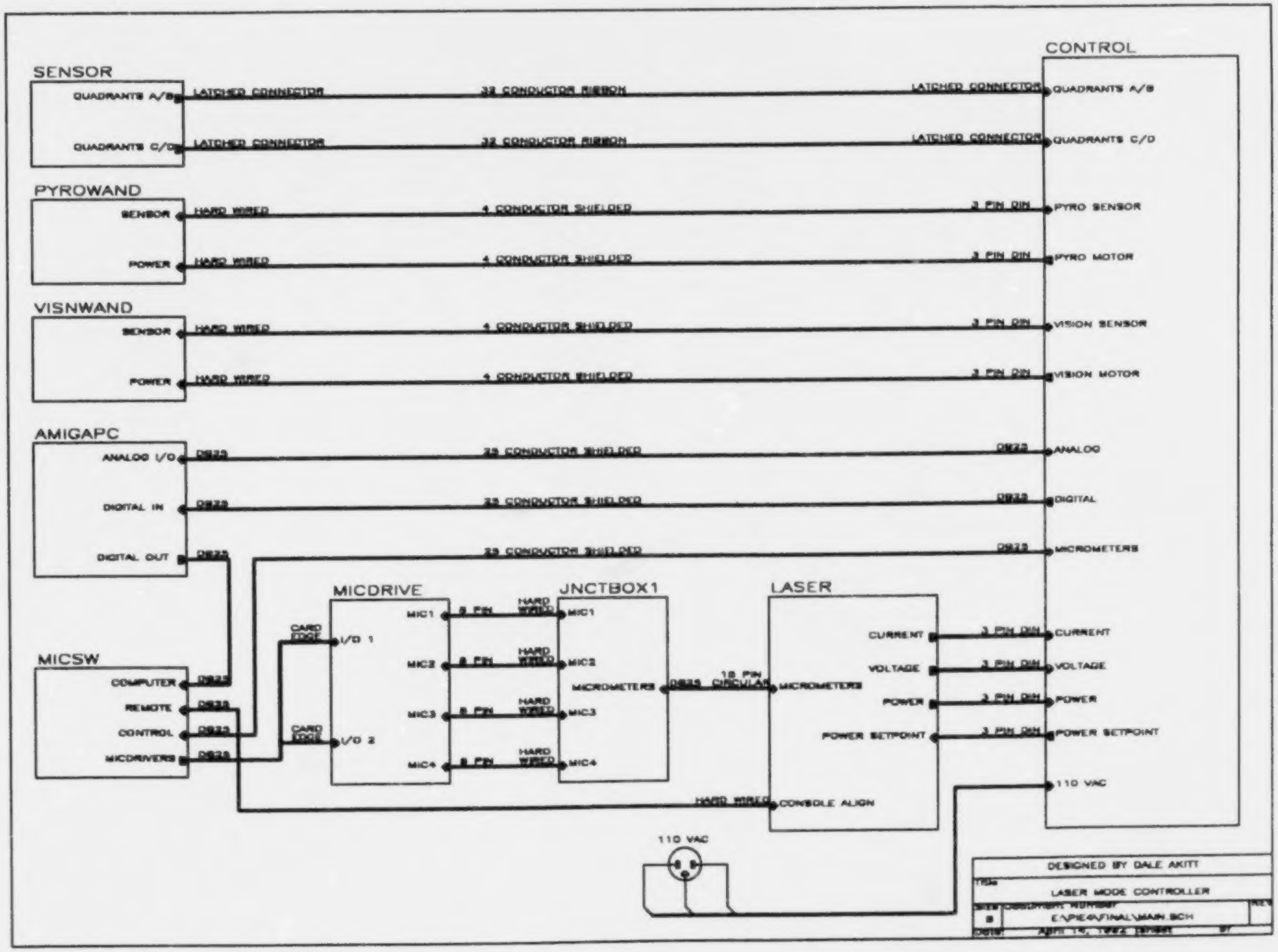
```
Model File: pie3\powersim.mod
Date:      3 /      2 /  1992
Time:      9 :     50
Timing:   0.0100000  ,DELTA  ;  38.0000     ,RANGE
PlotBlocks and Scales:
Format:
      BlockNo,   Plot-MINimum,   Plot-MAXimum;  Comment
Horz:      0 ,   0.0000      ,   30.0000     ;  Time
  Y1:     10 ,   0.0000      ,    5.0000     ;
  Y2:     11 ,   0.0000      ,    5.0000     ;
  Y3:     12 ,   0.0000      ,    5.0000     ;
  Y4:        ,               ,               ;


                         1 REM
;BLOCKS 2-11 SIMULATE THE OPEN-LOOP SYSTEM
   1.0000                2 CON                              ;1 SOURCE
   0.0000                3 CON                              ;0 SOURCE
   1.9000                4 CON
                         5 SUM       4     -11
   1.5000                6 PLS
  35.0000
   1.0000
                         7 IFE       6       29       3
   2.9900                8 INT       7
   0.5000000             9 BKL       8
   0.0000
   0.3036000            10 GAI       9
   1.0000               11 FIO      10
   1.3200
   0.9000000
                        20 REM
;CLOSED-LOOP CONTROLLER BEGINS HERE
   1.0000               21 CON                              ;LOGIC ONE SOURCE
   0.0000               22 CON                              ;LOGIC ZERO SOURCE
  25.0000               23 GAI       5                      ;CONTROLLER GAIN (Kc)
   1.0000               25 FIO      23
   0.9950000
  27.5000
                        26 ORR      47       48
                        27 ORR      71       28
                        28 ORR      57       58
                        29 IFE      25       27      31
                        30 ORR      47       48
  -1.0000               31 GAI      27
                        40 REM
;RELAY WITH DEAD-ZONE (SLOW CONTROLLER)
   1.0000               41 CON                              ;LOGIC ONE SOURCE
   0.0000               42 CON                              ;LOGIC ZERO SOURCE
   1.0000               43 GAI      25                      ;SIGNAL INPUT (SLOW)
   0.6000000            44 REL      41       42      42     ;HIGH RELAY
                                    43
  -0.6000000            45 REL      42       42      41     ;LOW RELAY
                                    43
   1.0000               46 GAI      21                      ;ENABLE INPUT
                        47 AND      44       46             ;HIGH O/P
                        48 AND      45       46             ;LOW O/P
                        50 REM
;RELAY WITH DEAD-ZONE (FAST CONTROLLER)
   1.0000               51 CON                              ;LOGIC ONE SOURCE
   0.0000               52 CON                              ;LOGIC ZERO SOURCE
   1.0000               53 GAI      23
   6.0000               54 REL      51       52      52     ;HIGH RELAY
                                    53
  -6.0000               55 REL      52       52      51     ;LOW RELAY
                                    53
   1.0000               56 GAI      30
                        57 AND      54       56             ;HIGH O/P
                        58 AND      55       56             ;LOW O/P
                        60 REM
;RESETTABLE PULSE GENERATOR
```

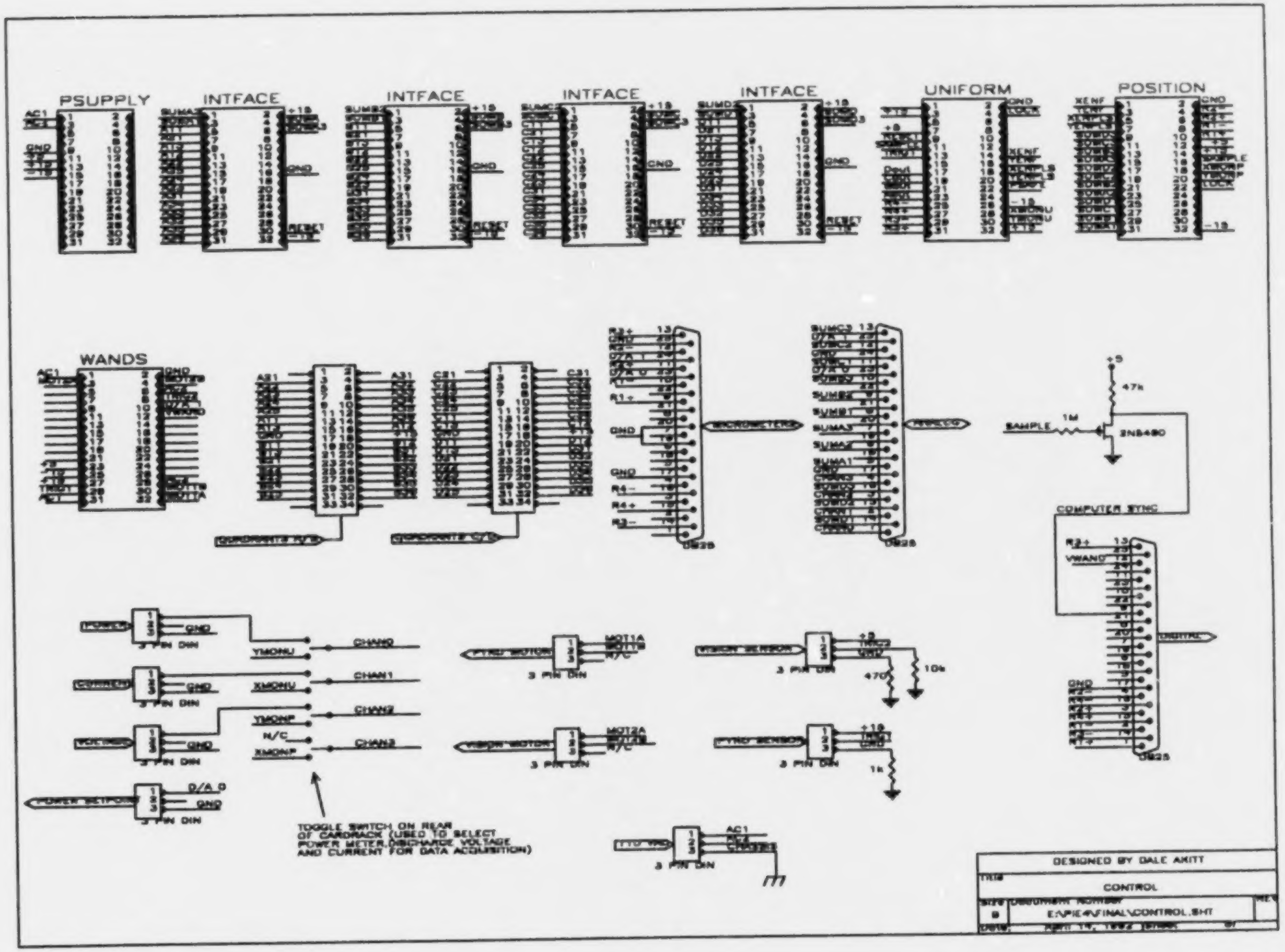
```
 1.0000         61 CON                              ;LOGIC ONE SOURCE
-1.0000         62 CON                              ;-1.0 SOURCE
 0.0000         63 CON                              ;LOGIC ZERO SOURCE
 6.6660         64 CON                              ;1/Ton (150 msec.)
 0.5000000      65 CON                              ;1/Toff (2.0 sec)
                66 IFE       70       64     -65
 0.0000         67 RIN       72       66
 1.0000         68 REL       61       63      63
                             67
 0.0000         69 REL       63       61      61
                             67
 1.0000         70 SRS       69       68
                71 AND       70       72            ;PULSE OUTPUT
                72 IFE       26       61      62    ;RESET/ENABLE INPUT
```

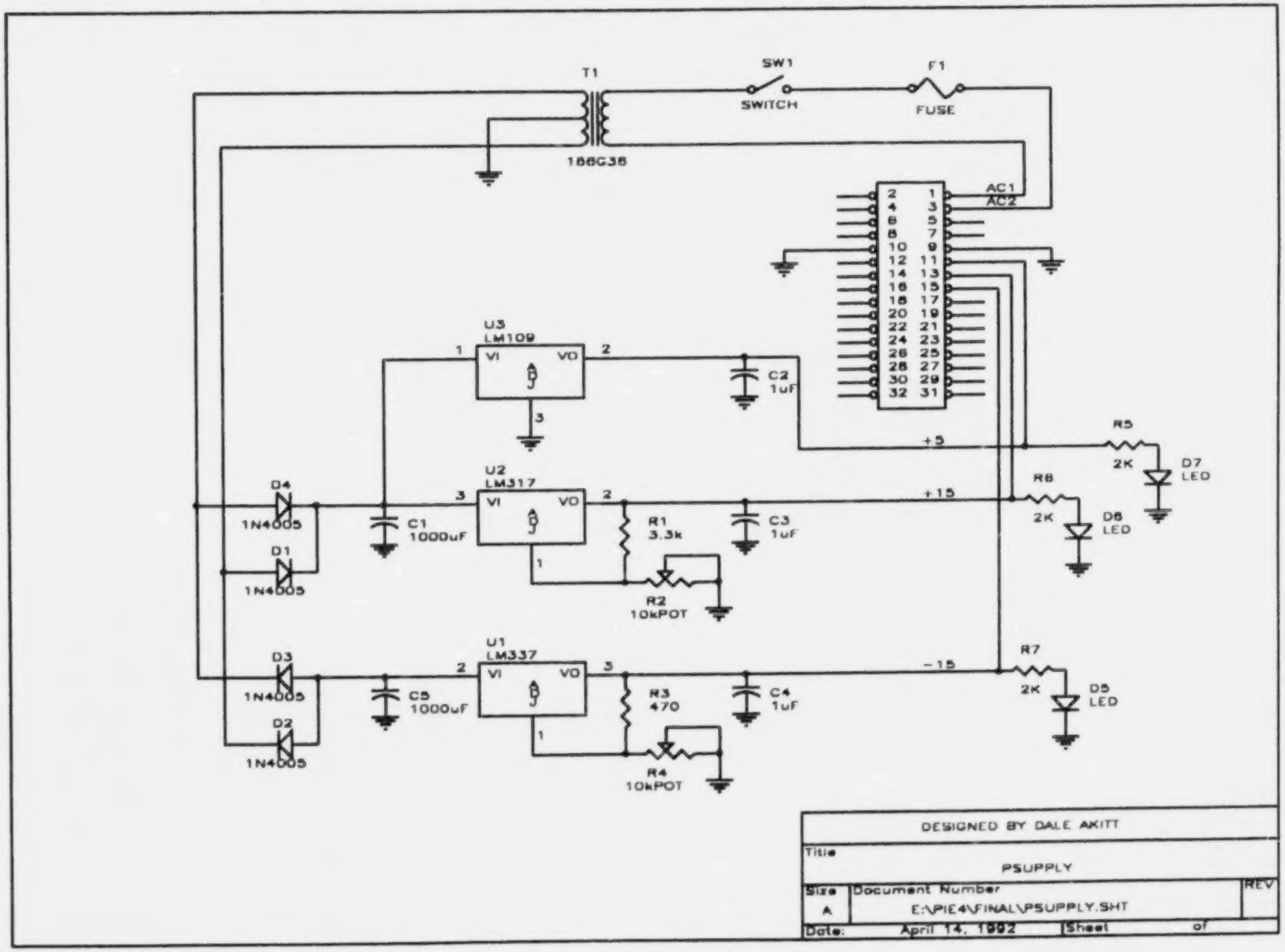
APPENDIX III - SCHEMATIC DRAWINGS

This appendix contains schematic drawings for the closed loop mode controller. Some of the drawings are quite large so a hierarchical sheet structure was utilized to enable presentation of the drawings on standard 8.5" X 11.0" paper. An attempt has been made to present the drawings in a top down order as follows:

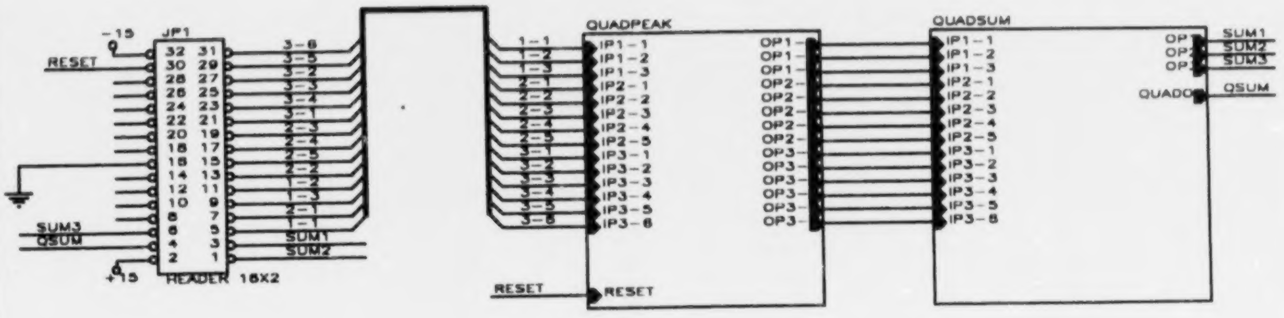
1. Master system block diagram.
2. Master connection diagram for the rear of the prototyping card rack.
3. DC Power Supply schematic
4. Detector interface circuit board (4 of these cards are present in the PIE-4 controller - 1 for each quadrant of the sensor array).
5. Beam uniformity controller board.
6. Beam position controller board.
7. Wand speed controller board (Includes circuits which control both the pyroelectric sensor and computer vision system wands).
8. External Connection Diagrams
   - Motor-micrometer connection schematic
   - Wand synchronization sensor schematic
   - Beam Profile Sensor array schematic
   - Amiga Microcomputer connections
   - PIE-4 power control connections
   - Monitor connections for discharge current and voltage, and laser power


SENSOR
QUADRANTS A/B
QUADRANTS C/D
LATCHED CONNECTOR
32 CONDUCTOR RIBBON
PYROWAND
SENSOR
POWER
HARD WIRED
4 CONDUCTOR SHIELDED
3 PIN DIN
VISNWAND
AMIGAPC
ANALOG I/O
DIGITAL IN
DIGITAL OUT
DB25
25 CONDUCTOR SHIELDED
MICSW
COMPUTER
REMOTE
CONTROL
MICDRIVERS
MICDRIVE
CARD EDGE
I/O 1
I/O 2
MIC1
MIC2
MIC3
MIC4
8 PIN
JNCTBOX1
MICROMETERS
18 PIN CIRCULAR
LASER
CURRENT
VOLTAGE
POWER
POWER SETPOINT
CONSOLE ALIGN
CONTROL
PYRO SENSOR
PYRO MOTOR
VISION SENSOR
VISION MOTOR
ANALOG
DIGITAL
110 VAC
DESIGNED BY DALE AKITT
LASER MODE CONTROLLER
E:\PIE4\FINAL\MAIN.SCH


PSUPPLY
INTFACE
INTFACE
INTFACE
INTFACE
UNIFORM
POSITION
WANDS
GND
+15
-15
RESET
XENF
SAMPLE
1M
+5
47k
2N5490
COMPUTER SYNC
MICROMETERS
DB25
R2+
VWAND
R1+
R4-
R4+
R3-
SUMC3
SUMB2
SUMB1
SUMA3
SUMA2
SUMA1
YMONU
XMONU
YMONP
N/C
XMONP
CHAN0
CHAN1
CHAN2
CHAN3
D/A 0
3 PIN DIN
MOT1A
MOT2A
+5
10k
470
+15
1k
AC1
TOGGLE SWITCH ON REAR OF CARDRACK (USED TO SELECT POWER METER,DISCHARGE VOLTAGE AND CURRENT FOR DATA ACQUISITION)
DESIGNED BY DALE AKITT
CONTROL
B
E:\PIE4\FINAL\CONTROL.SHT


T1
SW1
SWITCH
F1
FUSE
166G36
AC1
AC2
2 1
4 3
6 5
8 7
10 9
12 11
14 13
16 15
18 17
20 19
22 21
24 23
26 25
28 27
30 29
32 31
U3
LM109
VI
VO
1
2
3
C2
1uF
+5
R5
2K
D7
LED
U2
LM317
3
2
1
D4
1N4005
D1
1N4005
C1
1000uF
R1
3.3k
R2
10kPOT
C3
1uF
+15
R8
2K
D6
LED
U1
LM337
2
3
1
D3
1N4005
D2
1N4005
C5
1000uF
R3
470
R4
10kPOT
C4
1uF
-15
R7
2K
D5
LED
DESIGNED BY DALE AKITT
Title
PSUPPLY
Size
A
Document Number
E:\PIE4\FINAL\PSUPPLY.SHT
REV
Date:
April 14, 1992
Sheet
of

DETECTOR INTERFACE CIRCUIT BOARDS (QUADRANTS A THRU D)



DRAWN BY DALE AKITT

Title
INTFACE

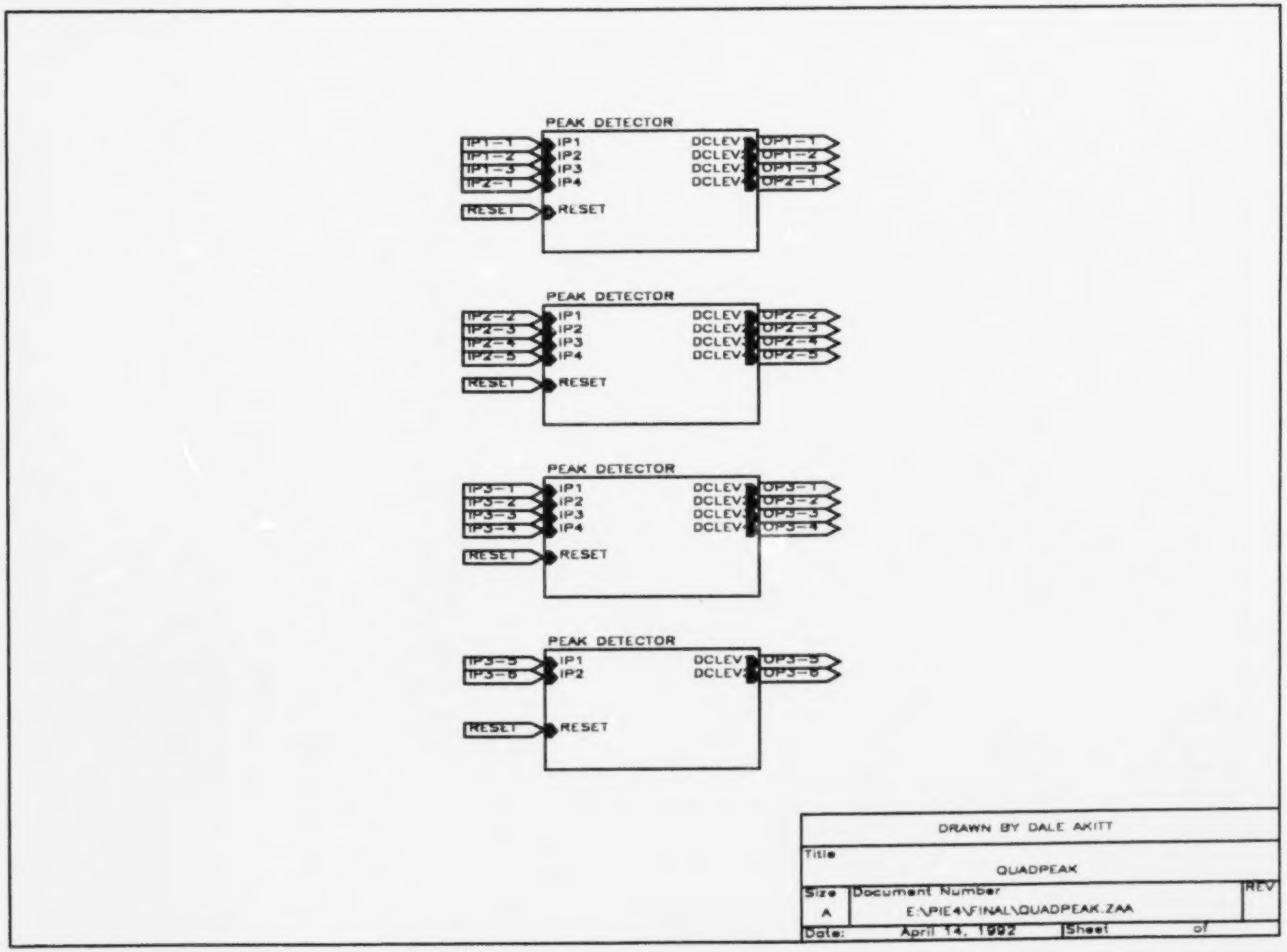
Size
A

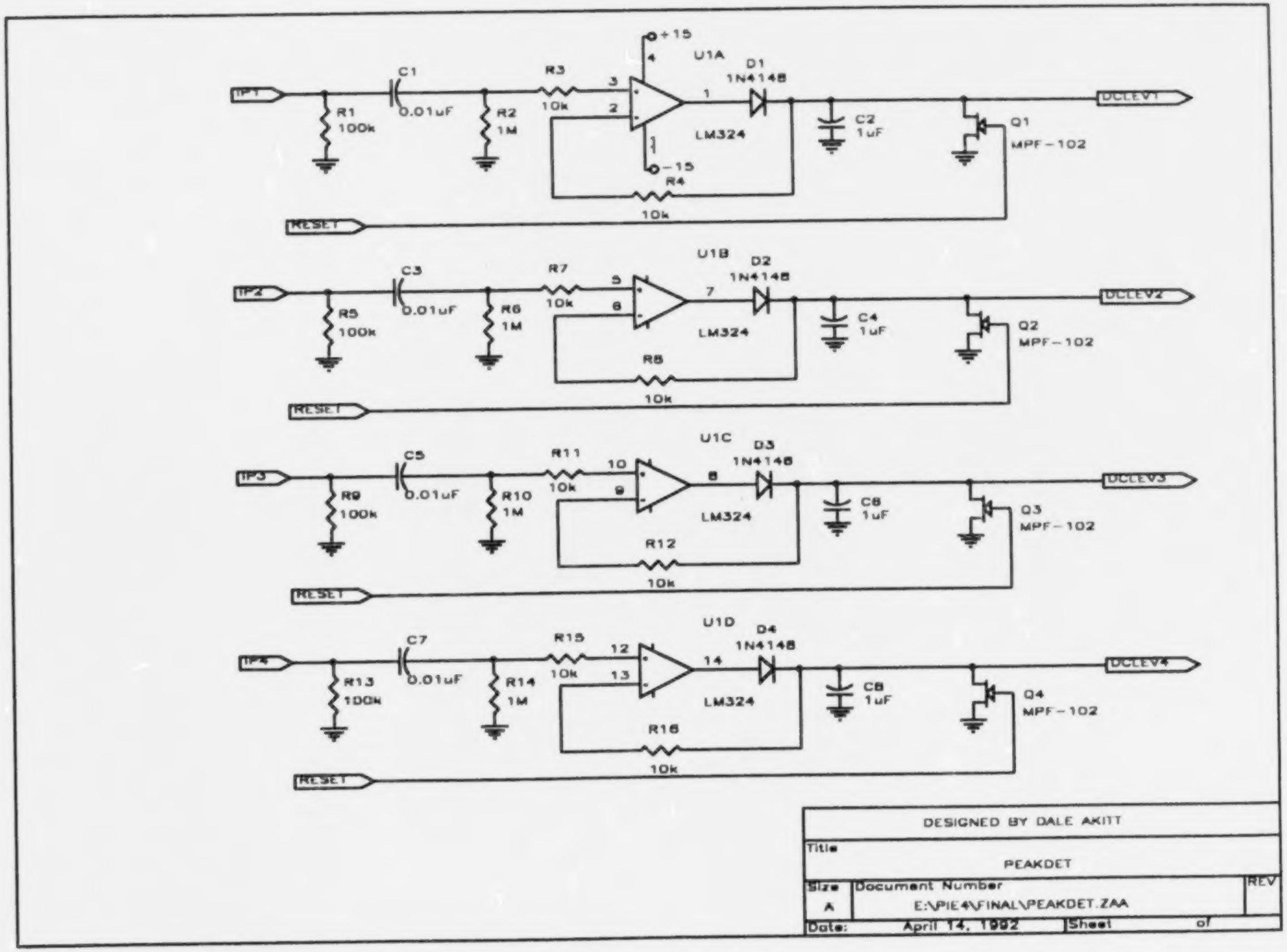
Document Number
E:\PIE4\FINAL\INTFACE.SCH

REV

Date: April 14, 1992

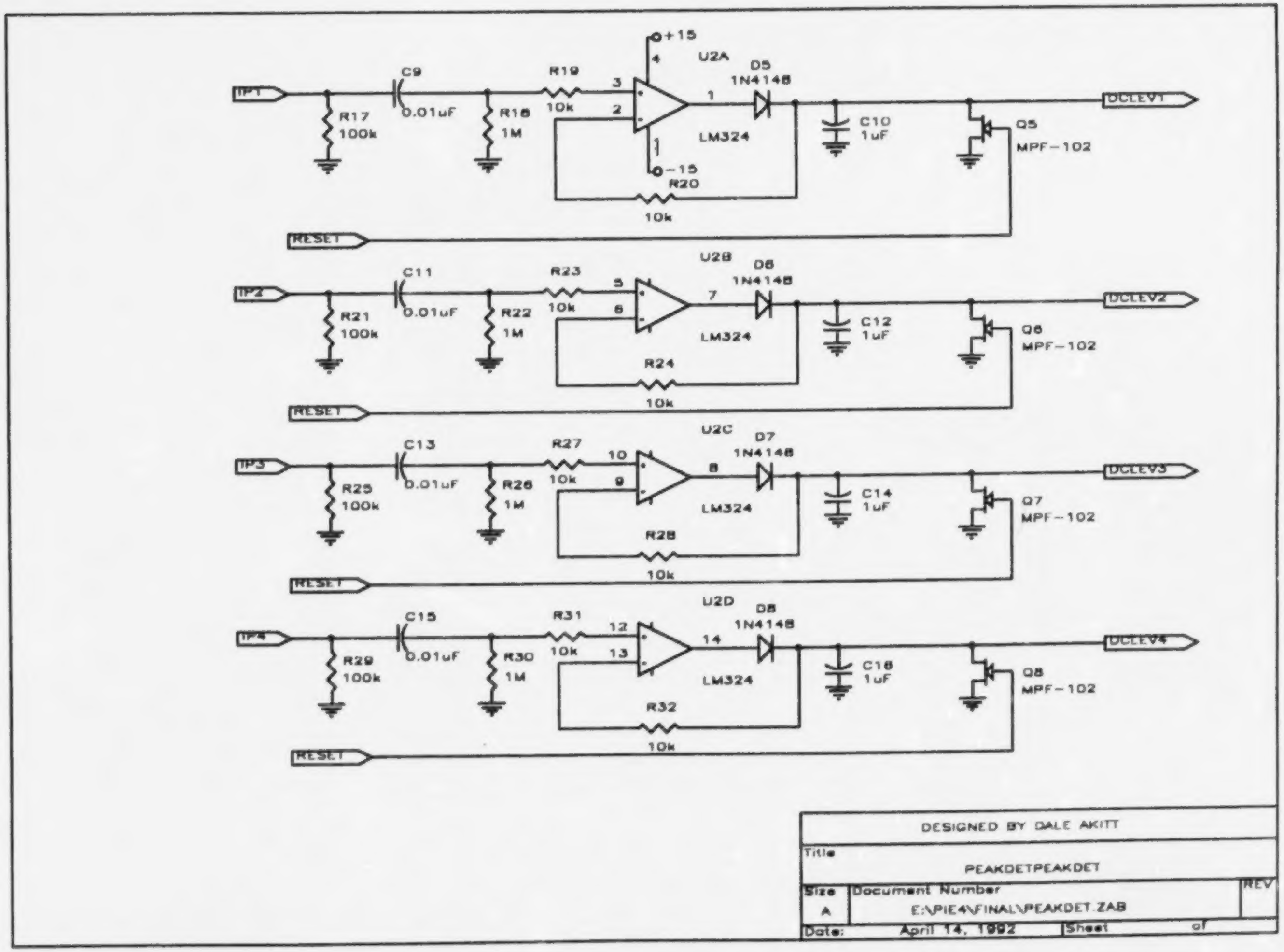
Sheet of

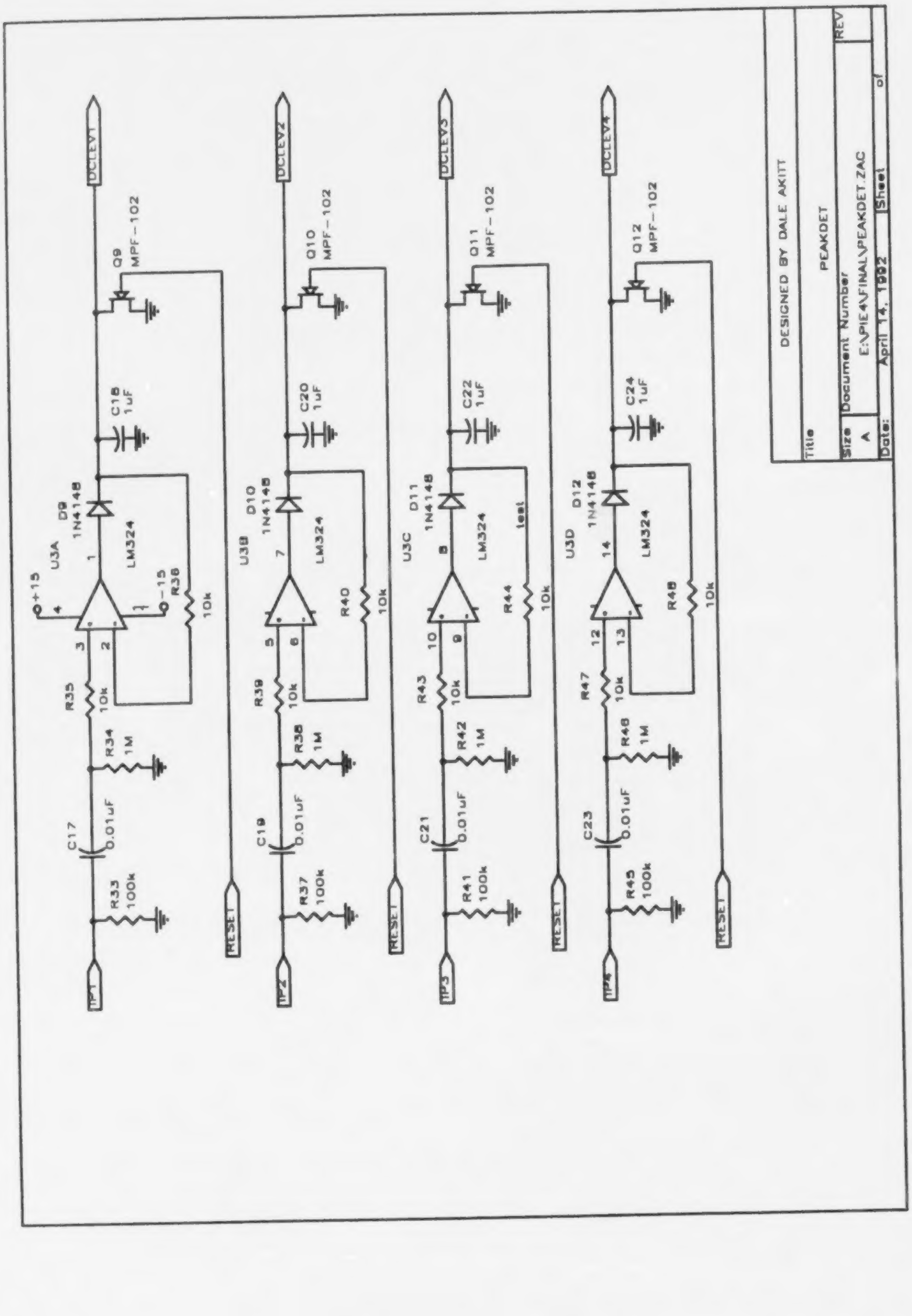

PEAK DETECTOR
IP1-1
IP1-2
IP1-3
IP2-1
RESET
IP1
IP2
IP3
IP4
RESET
DCLEV1
DCLEV2
DCLEV3
DCLEV4
OP1-1
OP1-2
OP1-3
OP2-1
PEAK DETECTOR
IP2-2
IP2-3
IP2-4
IP2-5
RESET
IP1
IP2
IP3
IP4
RESET
DCLEV1
DCLEV2
DCLEV3
DCLEV4
OP2-2
OP2-3
OP2-4
OP2-5
PEAK DETECTOR
IP3-1
IP3-2
IP3-3
IP3-4
RESET
IP1
IP2
IP3
IP4
RESET
DCLEV1
DCLEV2
DCLEV3
DCLEV4
OP3-1
OP3-2
OP3-3
OP3-4
PEAK DETECTOR
IP3-5
IP3-6
RESET
IP1
IP2
RESET
DCLEV1
DCLEV2
OP3-5
OP3-6
DRAWN BY DALE AKITT
Title
QUADPEAK
Size
A
Document Number
E:\PIE4\FINAL\QUADPEAK.ZAA
REV
Date:
April 14, 1992
Sheet
of

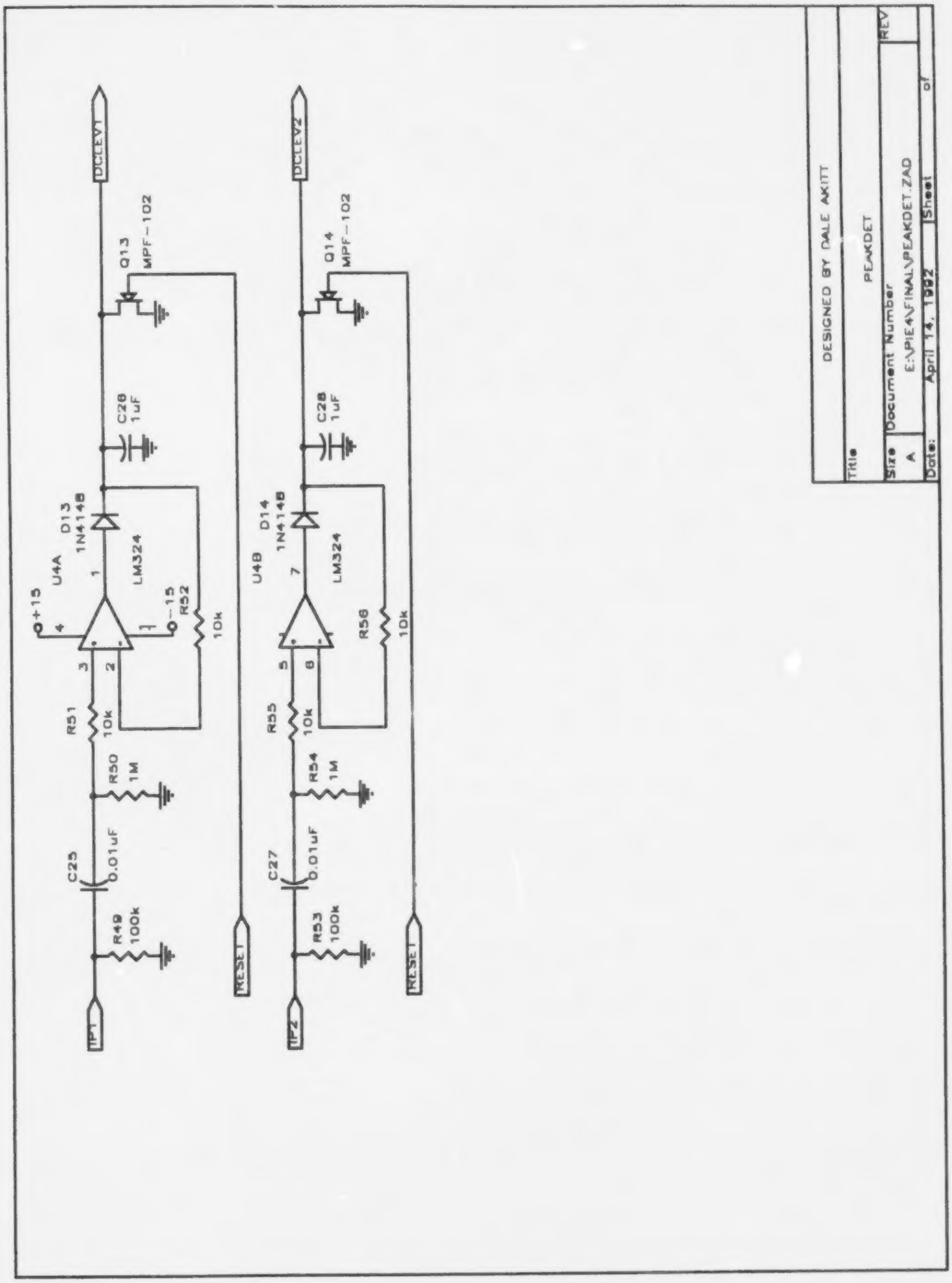


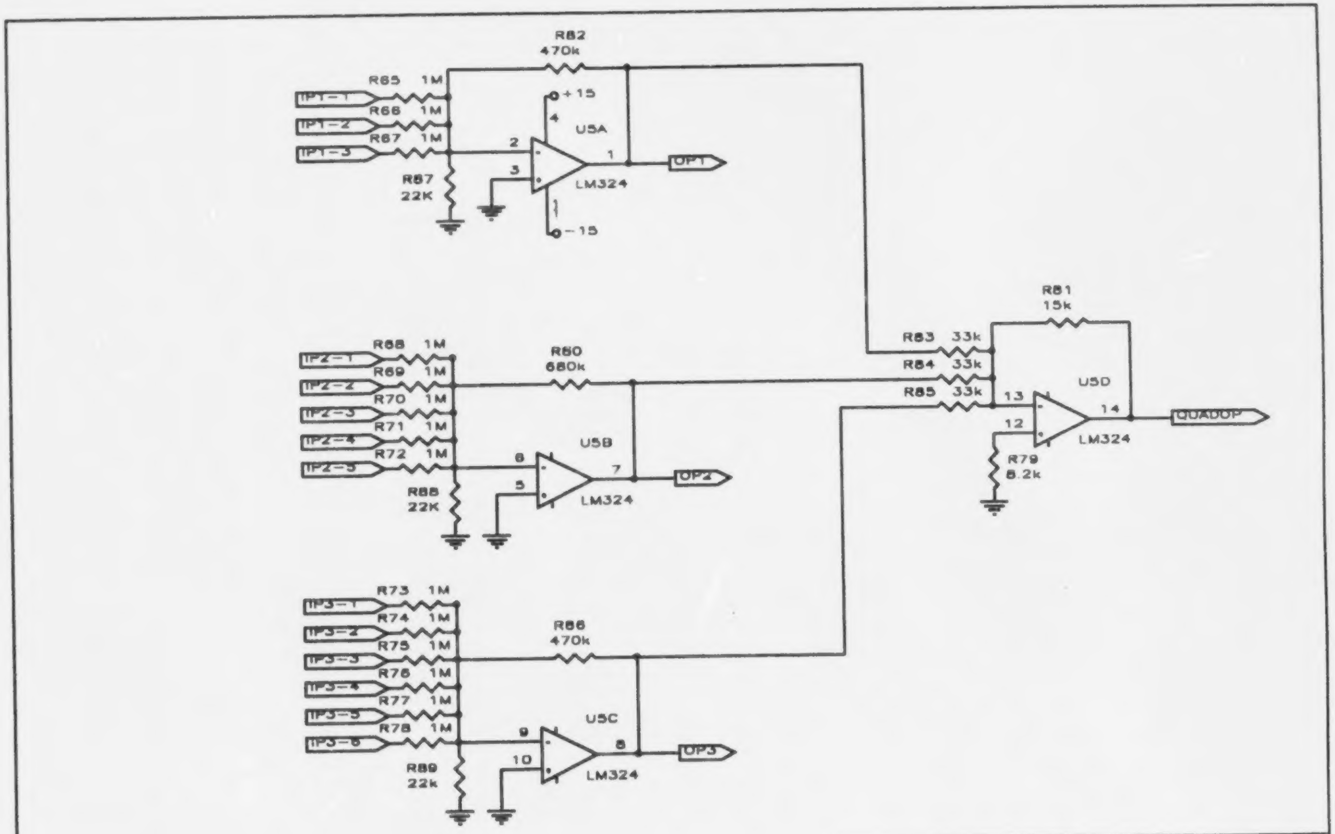
DESIGNED BY DALE AKITT

Title	PEAKDET	
Size A	Document Number E:\PIE4\FINAL\PEAKDET.ZAA	REV
Date:	April 14, 1992	Sheet of

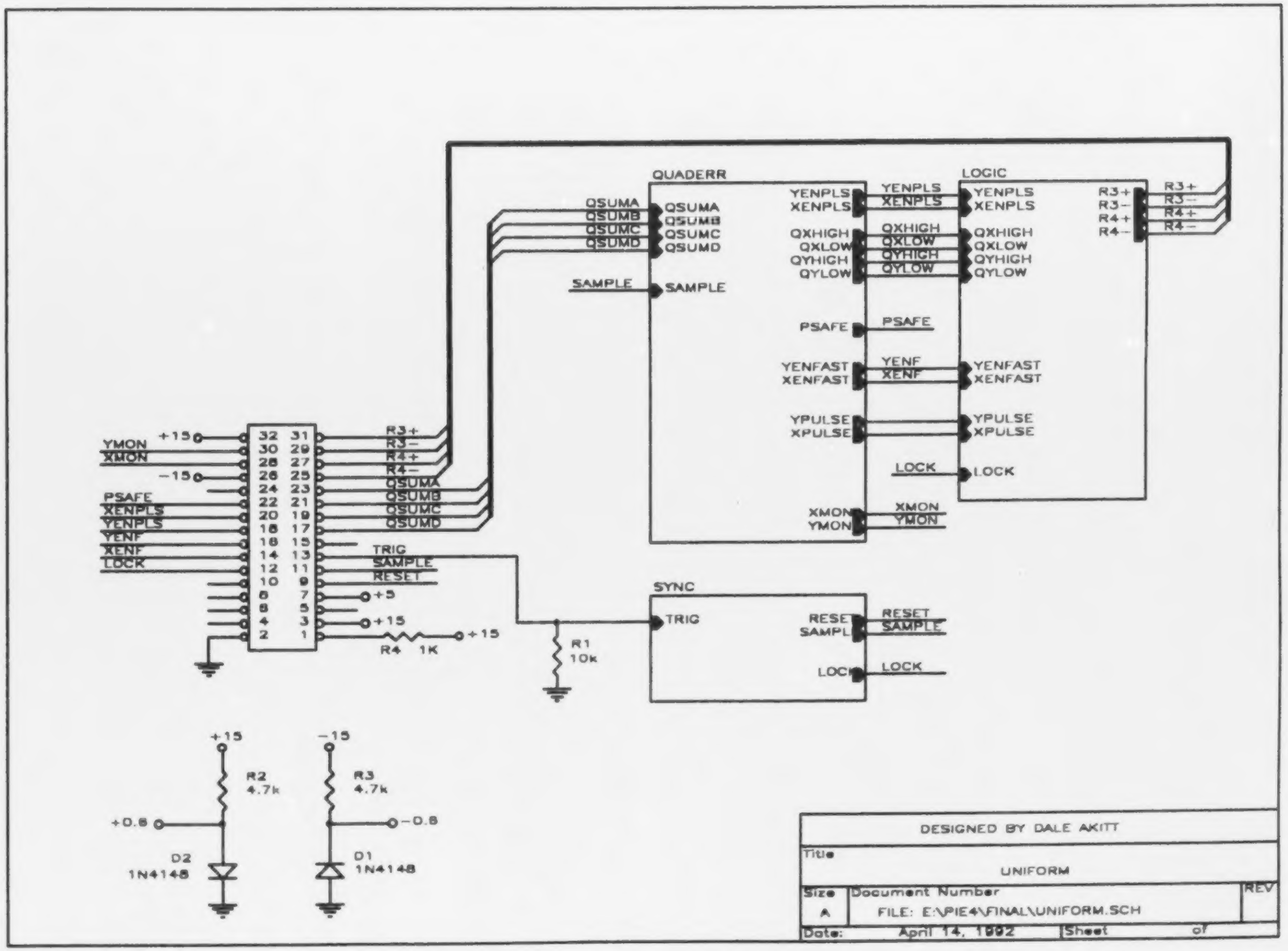

IP1
R17
100k
C9
0.01uF
R18
1M
R19
10k
3
2
4
+15
-15
U2A
1
D5
1N4148
LM324
R20
10k
C10
1uF
Q5
MPF-102
DCLEV1
RESET
IP2
R21
100k
C11
0.01uF
R22
1M
R23
10k
5
6
U2B
7
D6
1N4148
LM324
R24
10k
C12
1uF
Q6
MPF-102
DCLEV2
RESET
IP3
R25
100k
C13
0.01uF
R26
1M
R27
10k
10
9
U2C
8
D7
1N4148
LM324
R28
10k
C14
1uF
Q7
MPF-102
DCLEV3
RESET
IP4
R29
100k
C15
0.01uF
R30
1M
R31
10k
12
13
U2D
14
D8
1N4148
LM324
R32
10k
C16
1uF
Q8
MPF-102
DCLEV4
RESET
DESIGNED BY DALE AKITT
Title
PEAKDETPEAKDET
Size
A
Document Number
E:\PIE4\FINAL\PEAKDET.ZAB
REV
Date:
April 14, 1992
Sheet
of


IP1
R33
100k
C17
0.01uF
R34
1M
R35
10k
3
2
4
+15
U3A
1
LM324
11
-15
R36
10k
D9
1N4148
C18
1uF
Q9
MPF-102
DCLEV1
RESET
IP2
R37
100k
C19
0.01uF
R38
1M
R39
10k
5
6
U3B
7
LM324
R40
10k
D10
1N4148
C20
1uF
Q10
MPF-102
DCLEV2
RESET
IP3
R41
100k
C21
0.01uF
R42
1M
R43
10k
10
9
U3C
8
LM324
R44
10k
test
D11
1N4148
C22
1uF
Q11
MPF-102
DCLEV3
RESET
IP4
R45
100k
C23
0.01uF
R46
1M
R47
10k
12
13
U3D
14
LM324
R48
10k
D12
1N4148
C24
1uF
Q12
MPF-102
DCLEV4
RESET
DESIGNED BY DALE AKITT
Title
PEAKDET
Size
A
Document Number
E:\PIE4\FINAL\PEAKDET.ZAC
REV
Date:
April 14, 1992
Sheet
of


DESIGNED BY DALE AKITT
Title
PEAKDET
Size
A
Document Number
E:\PIE4\FINAL\PEAKDET.ZAD
REV
Date:
April 14, 1992
Sheet
of
IP1
R49
100k
C25
0.01uF
R50
1M
R51
10k
3
2
4
+15
-15
U4A
LM324
1
R52
10k
D13
1N4148
C28
1uF
Q13
MPF-102
DCLEV1
RESET
IP2
R53
100k
C27
0.01uF
R54
1M
R55
10k
5
6
U4B
LM324
7
R56
10k
D14
1N4148
C28
1uF
Q14
MPF-102
DCLEV2
RESET



DESIGNED BY DALE AKITT			
Title QUADSUM			
Size A	Document Number E:\PIE4\FINAL\QUADSUM.ZAA		REV
Date:	April 14, 1992	Sheet	of

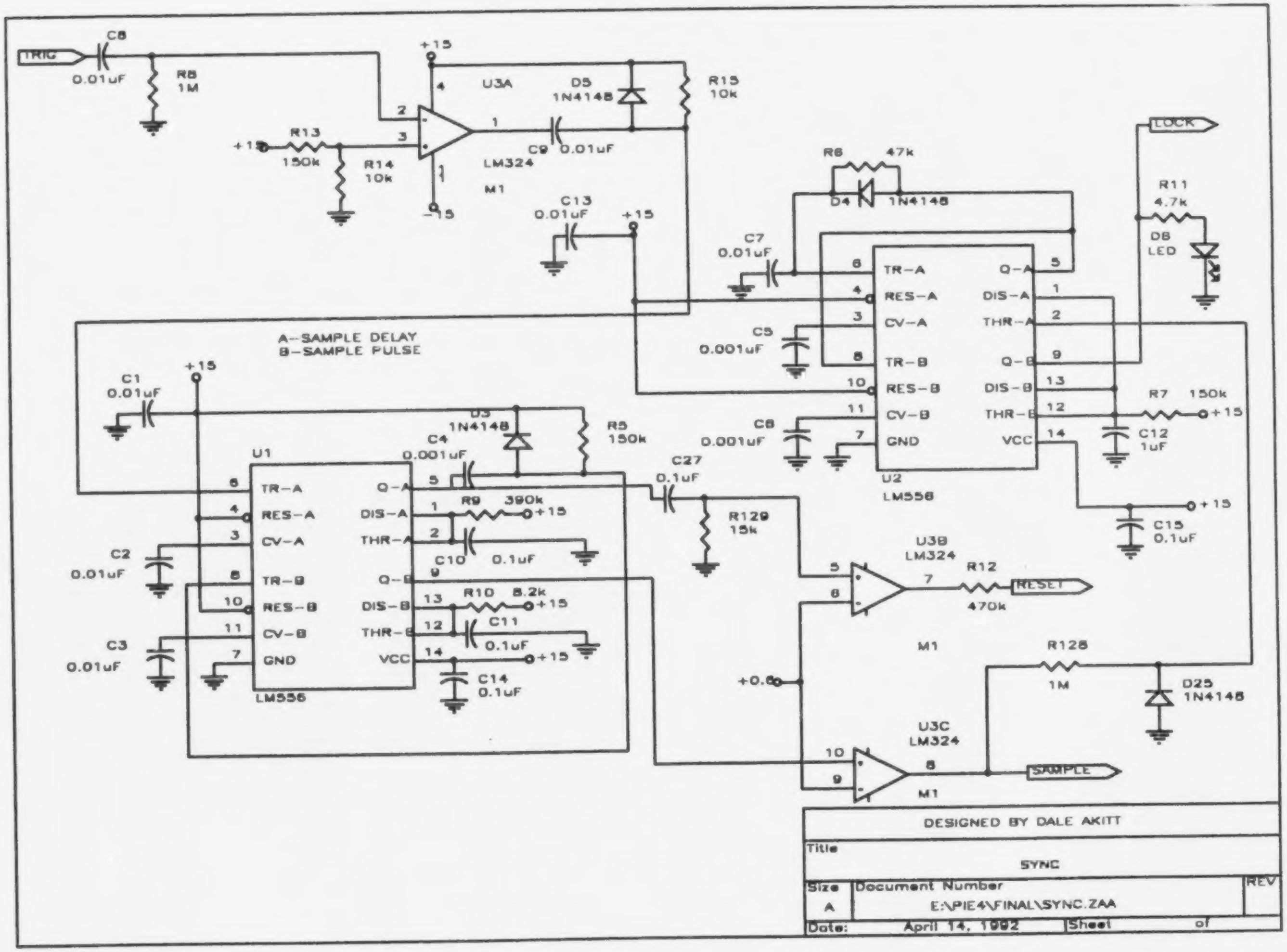


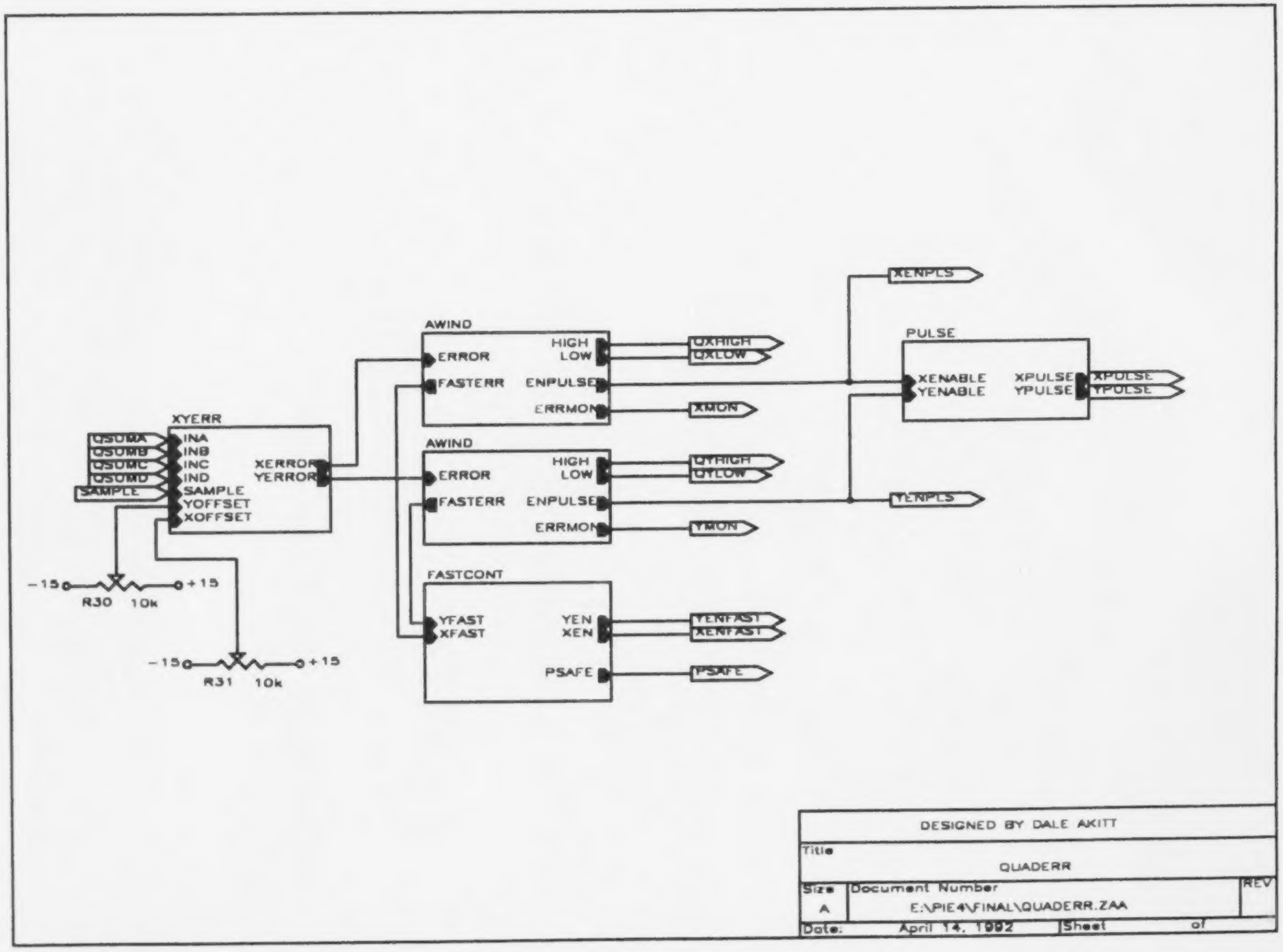
DESIGNED BY DALE AKITT

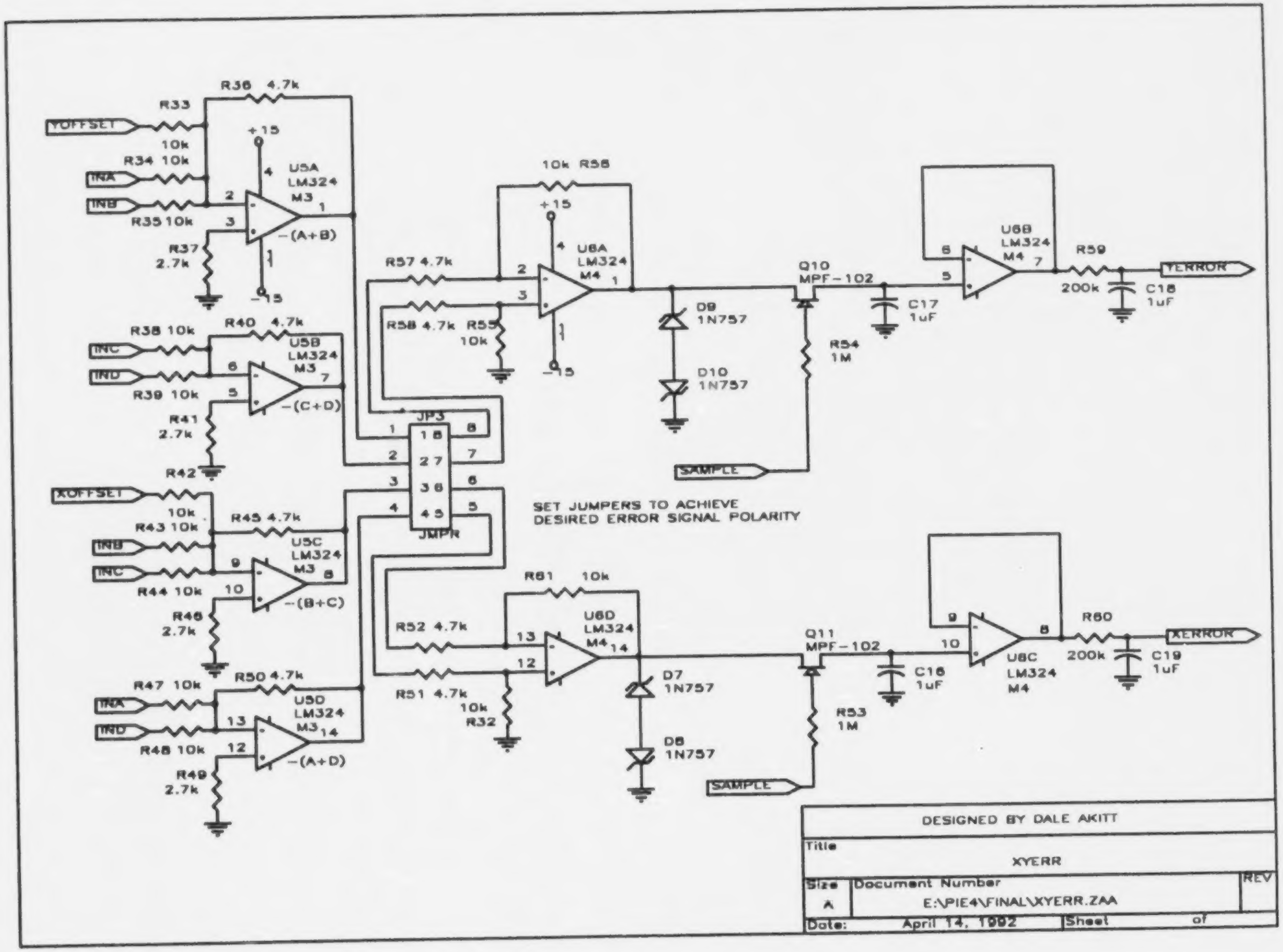
Title
UNIFORM

Size A
Document Number
FILE: E:\PIE4\FINAL\UNIFORM.SCH
REV

Date: April 14, 1992
Sheet of


TRIG
C8
0.01uF
R8
1M
+15
4
U3A
D5
1N4148
R15
10k
2
3
1
+15
R13
150k
R14
10k
LM324
C9 0.01uF
M1
-15
LOCK
R6 47k
D4 1N4148
R11
4.7k
D8
LED
C13
0.01uF
+15
C7
0.01uF
TR-A
RES-A
CV-A
TR-B
RES-B
CV-B
GND
Q-A
DIS-A
THR-A
Q-B
DIS-B
THR-B
VCC
C5
0.001uF
C6
0.001uF
U2
LM556
R7 150k
+15
C12
1uF
C15
0.1uF
+15
A-SAMPLE DELAY
B-SAMPLE PULSE
C1
0.01uF
+15
D3
1N4148
R5
150k
C4
0.001uF
C27
0.1uF
U1
R9 390k
+15
C10 0.1uF
R10 8.2k
+15
C11
0.1uF
+15
C14
0.1uF
C2
0.01uF
C3
0.01uF
LM556
R129
15k
U3B
LM324
R12
470k
RESET
M1
+0.6
R128
1M
D25
1N4148
U3C
LM324
SAMPLE
M1
DESIGNED BY DALE AKITT
Title
SYNC
Size
A
Document Number
E:\PIE4\FINAL\SYNC.ZAA
REV
Date:
April 14, 1992
Sheet
of


XYERR
QSUMA
QSUMB
QSUMC
QSUMD
SAMPLE
INA
INB
INC
IND
SAMPLE
YOFFSET
XOFFSET
XERROR
YERROR
-15
R30
10k
+15
-15
R31
10k
+15
AWIND
ERROR
FASTERR
HIGH
LOW
ENPULSE
ERRMON
QXHIGH
QXLOW
XMON
AWIND
ERROR
FASTERR
HIGH
LOW
ENPULSE
ERRMON
QYHIGH
QYLOW
YMON
FASTCONT
YFAST
XFAST
YEN
XEN
PSAFE
YENFAST
XENFAST
PSAFE
XENPLS
PULSE
XENABLE
YENABLE
XPULSE
YPULSE
XPULSE
YPULSE
YENPLS
DESIGNED BY DALE AKITT
Title
QUADERR
Size
A
Document Number
E:\PIE4\FINAL\QUADERR.ZAA
REV
Date:
April 14, 1992
Sheet
of



SET JUMPERS TO ACHIEVE
DESIRED ERROR SIGNAL POLARITY

DESIGNED BY DALE AKITT

Title XYERR

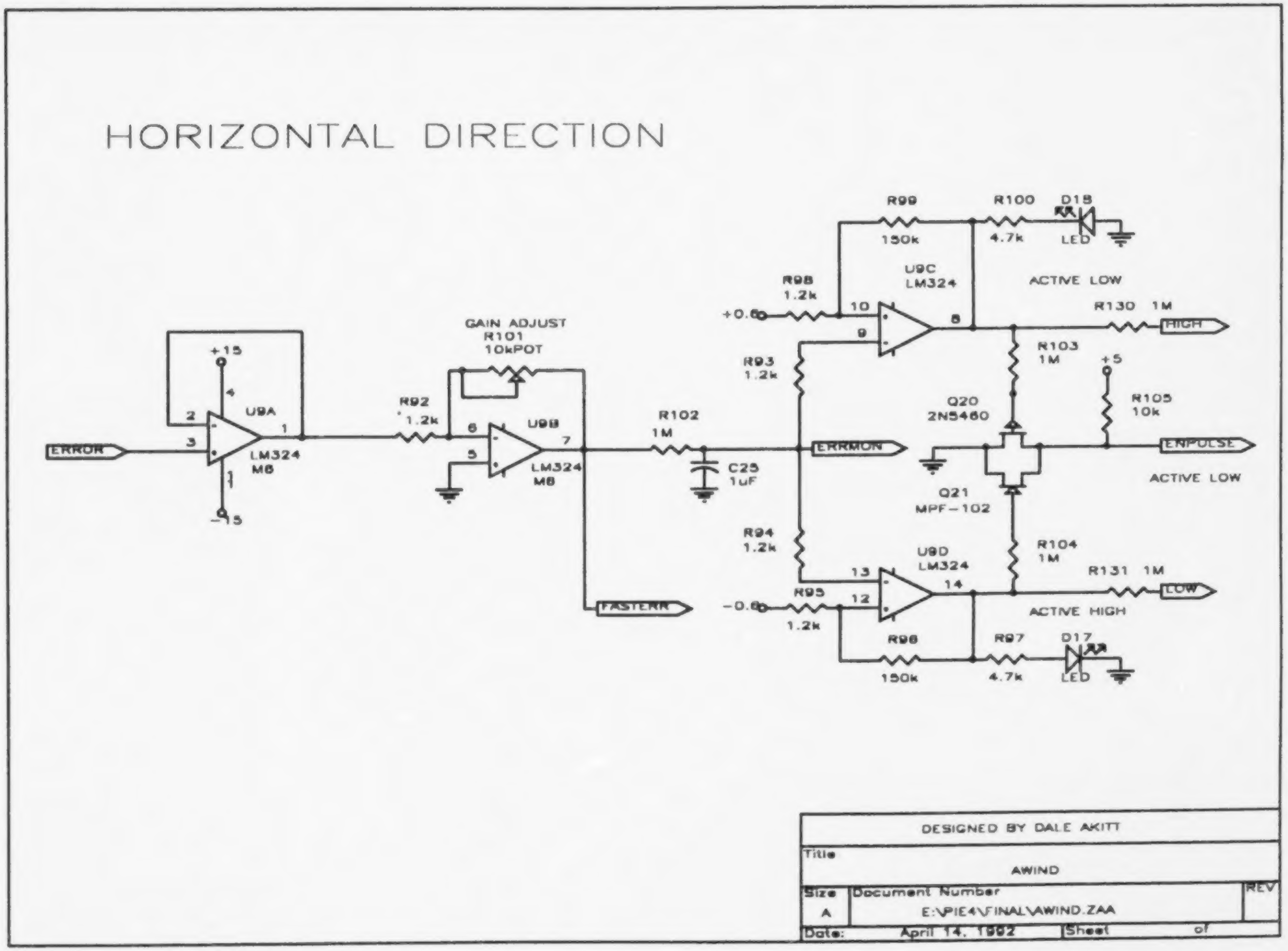
Size A

Document Number E:\PIE4\FINAL\XYERR.ZAA

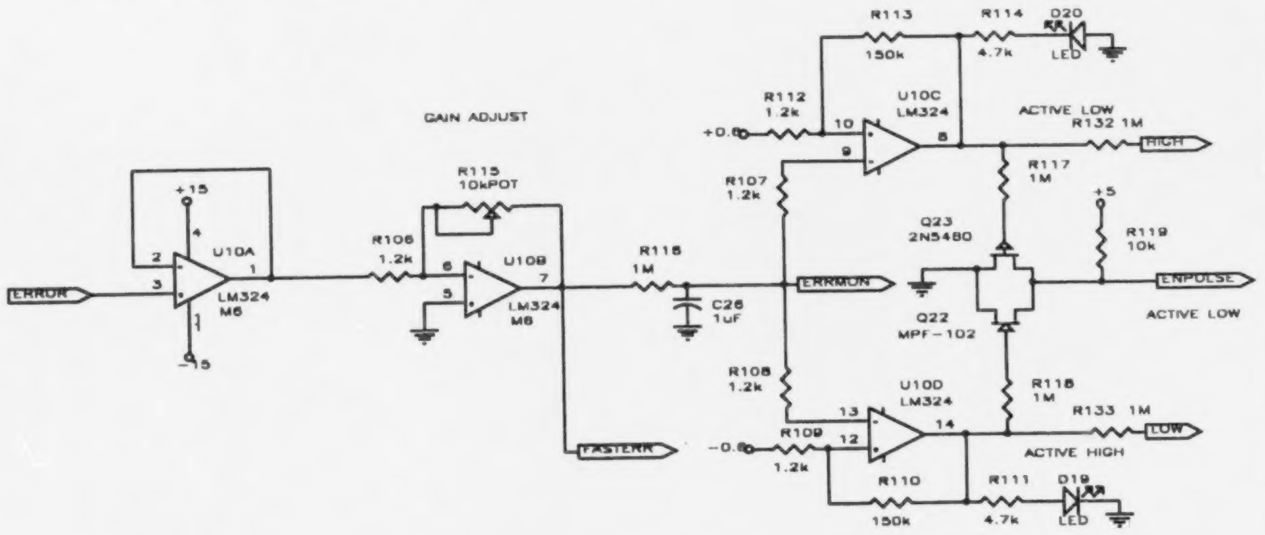
REV

Date: April 14, 1992

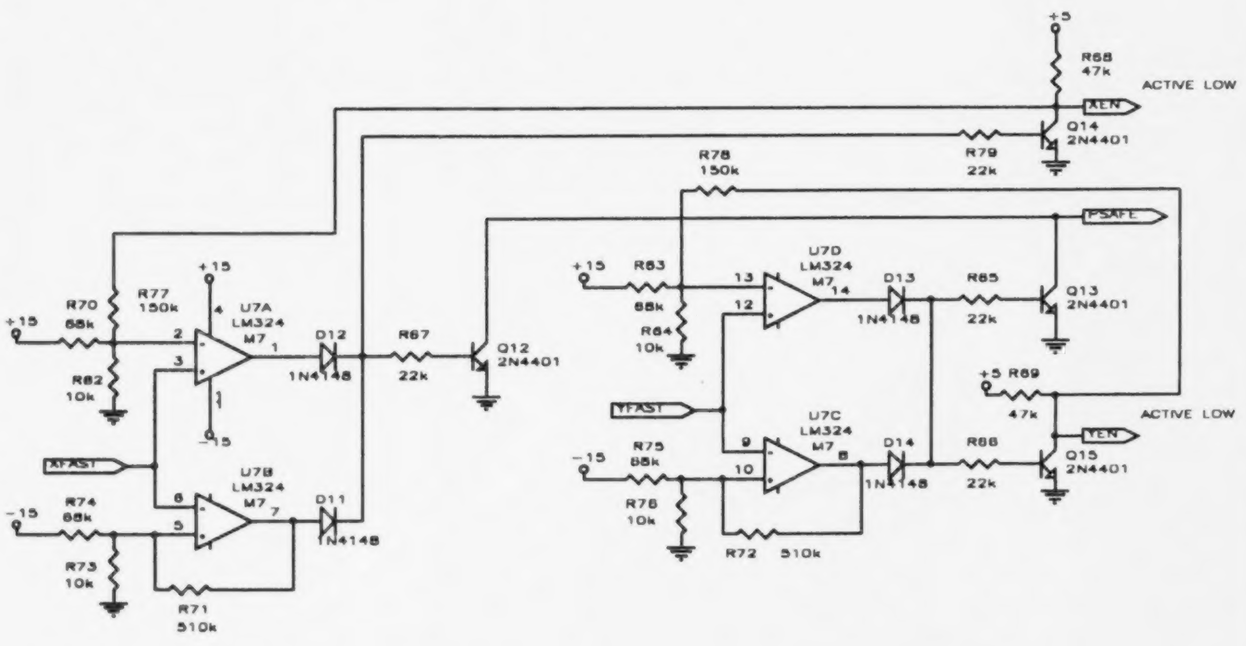
Sheet of


HORIZONTAL DIRECTION
ERROR
+15
-15
2
3
4
1
U9A
LM324
M8
R92
1.2k
GAIN ADJUST
R101
10kPOT
6
5
7
U9B
LM324
M8
FASTERR
R102
1M
C25
1uF
+0.6
-0.6
R98
1.2k
R93
1.2k
R94
1.2k
R95
1.2k
R99
150k
R100
4.7k
D18
LED
U9C
LM324
10
9
8
ACTIVE LOW
R130 1M
HIGH
R103
1M
+5
R105
10k
Q20
2N5460
Q21
MPF-102
ERRMON
ENPULSE
ACTIVE LOW
R104
1M
U9D
LM324
13
12
14
R131 1M
LOW
ACTIVE HIGH
R96
150k
R97
4.7k
D17
LED
DESIGNED BY DALE AKITT
Title
AWIND
Size
A
Document Number
E:\PIE4\FINAL\AWIND.ZAA
REV
Date:
April 14, 1992
Sheet
of

VERTICAL DIRECTION

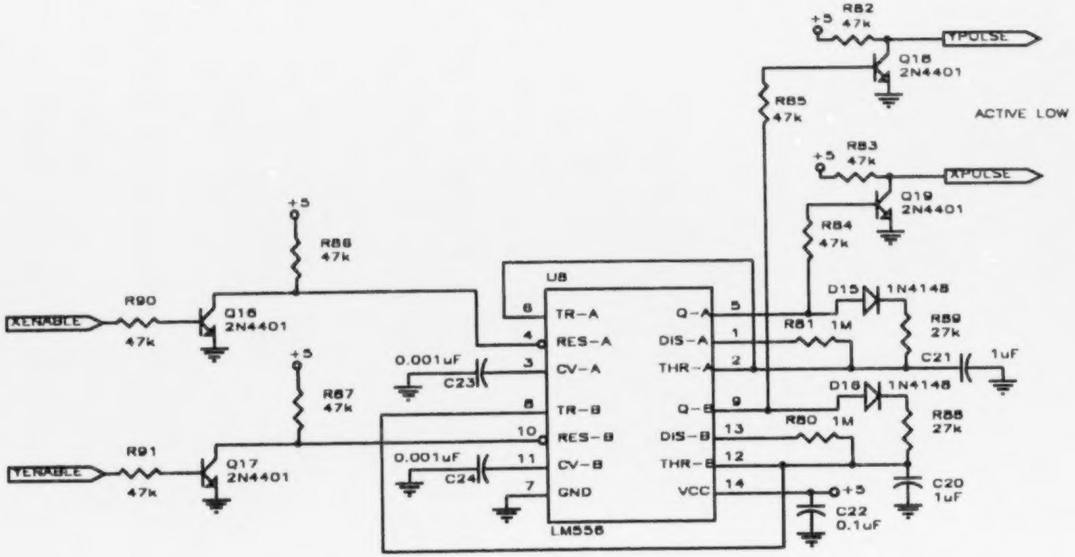


DESIGNED BY DALE AKITT
Title
AWIND
Size A
Document Number E:\PIE4\FINAL\AWIND.ZAB
REV
Date: April 14, 1992
Sheet of



DESIGNED BY DALE AKITT

Title	FASTCONT	
Size A	Document Number E:\PIE4\FINAL\FASTCONT.ZAA	REV
Date:	April 14, 1992	Sheet of



T-ON = 0.693•R•Ct (R IN SERIES WITH D15, D16)
T-OFF = 0.693•R•Ct (R AT PINS 1 AND 13)

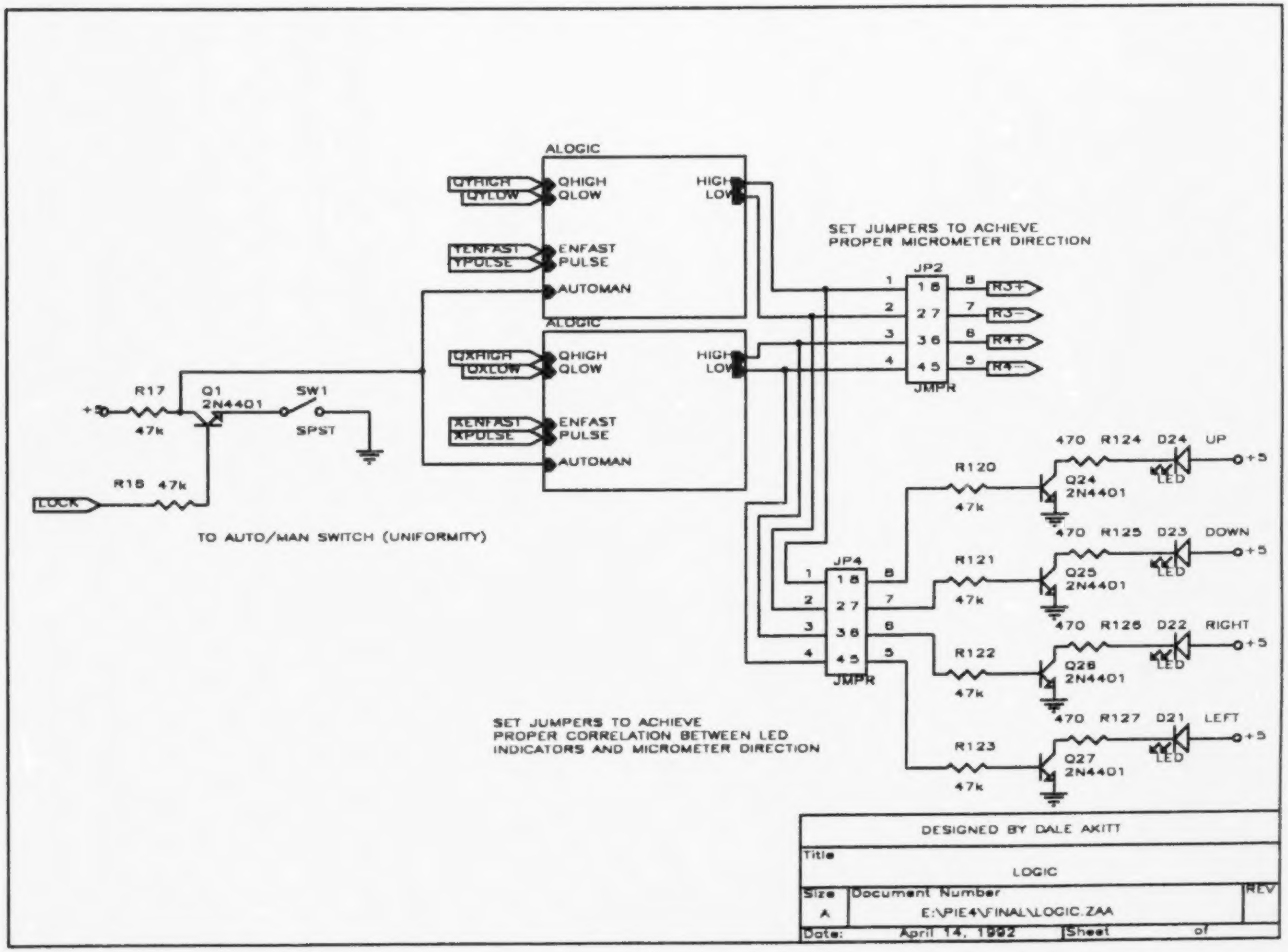
AS SHOWN T-ON = 18.7 ms. T-OFF = 0.7 sec.

DESIGNED BY DALE AKITT

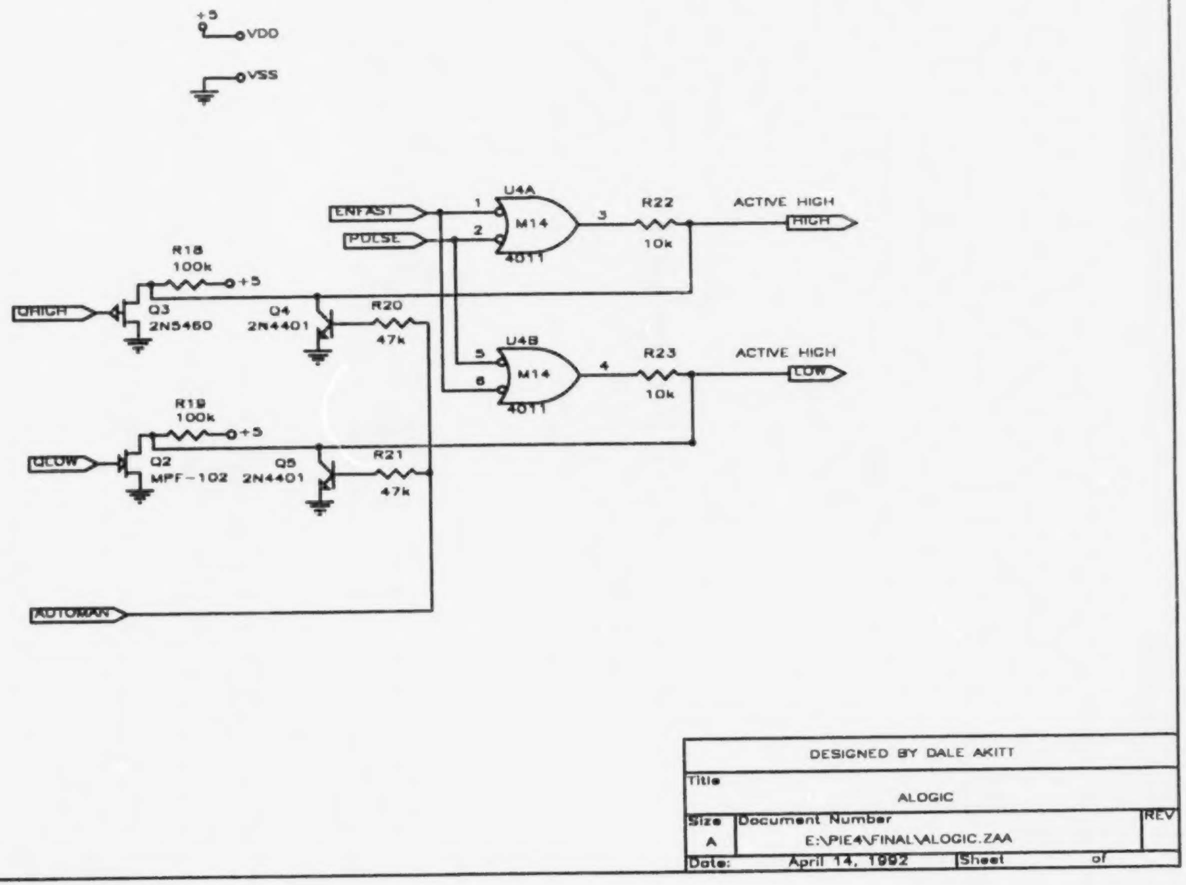
Title
PULSE

Size A
Document Number E:\PIE4\FINAL\PULSE.ZAA
REV

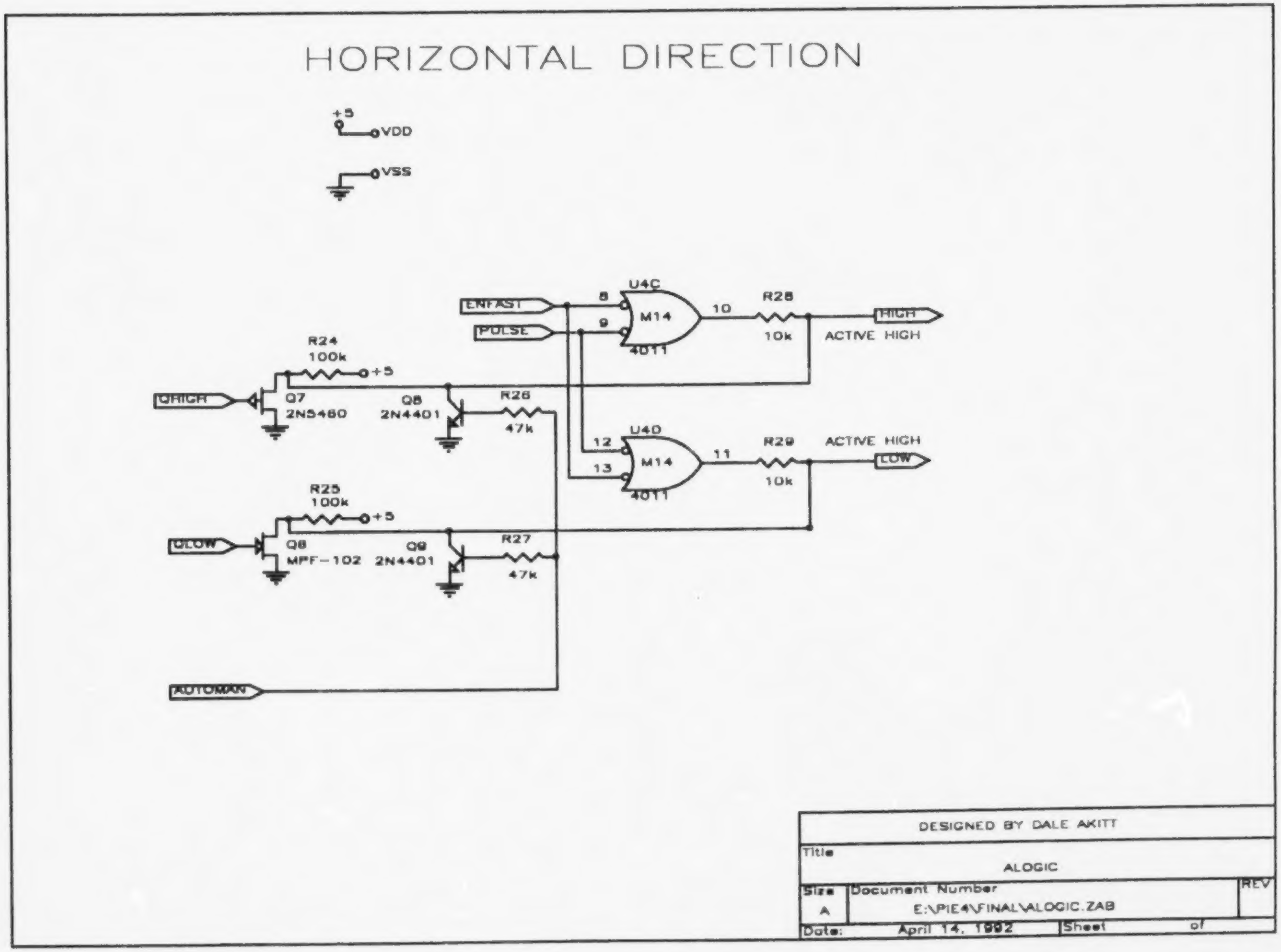
Date: April 14, 1992
Sheet of


ALOGIC
QYHIGH
QYLOW
QHIGH
QLOW
HIGH
LOW
TENFAST
YPULSE
ENFAST
PULSE
AUTOMAN
ALOGIC
QXHIGH
QXLOW
QHIGH
QLOW
HIGH
LOW
XENFAST
XPULSE
ENFAST
PULSE
AUTOMAN
SET JUMPERS TO ACHIEVE
PROPER MICROMETER DIRECTION
JP2
1 2 3 4
18 27 36 45
8 7 6 5
R3+
R3−
R4+
R4−
JMPR
+5
R17
47k
Q1
2N4401
SW1
SPST
R18 47k
LOCK
TO AUTO/MAN SWITCH (UNIFORMITY)
JP4
1 2 3 4
18 27 36 45
8 7 6 5
JMPR
R120
47k
R121
47k
R122
47k
R123
47k
470 R124 D24 UP
Q24
2N4401
LED
+5
470 R125 D23 DOWN
Q25
2N4401
LED
+5
470 R126 D22 RIGHT
Q26
2N4401
LED
+5
470 R127 D21 LEFT
Q27
2N4401
LED
+5
SET JUMPERS TO ACHIEVE
PROPER CORRELATION BETWEEN LED
INDICATORS AND MICROMETER DIRECTION
DESIGNED BY DALE AKITT
Title
LOGIC
Size
A
Document Number
E:\PIE4\FINAL\LOGIC.ZAA
REV
Date:
April 14, 1992
Sheet
of

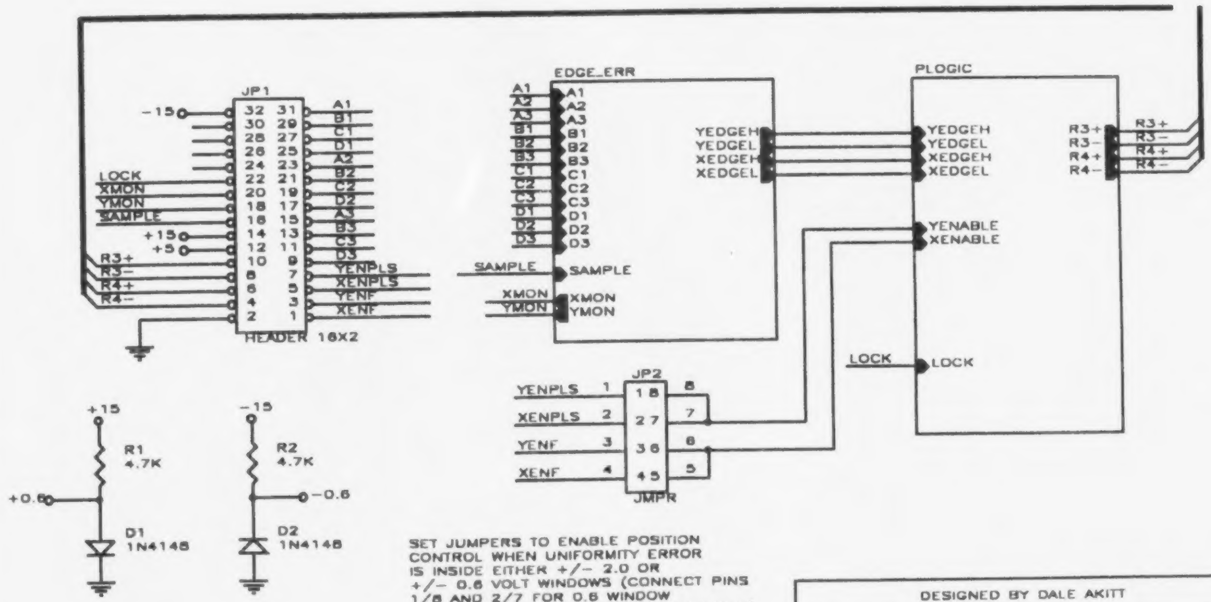
VERTICAL DIRECTION



HORIZONTAL DIRECTION

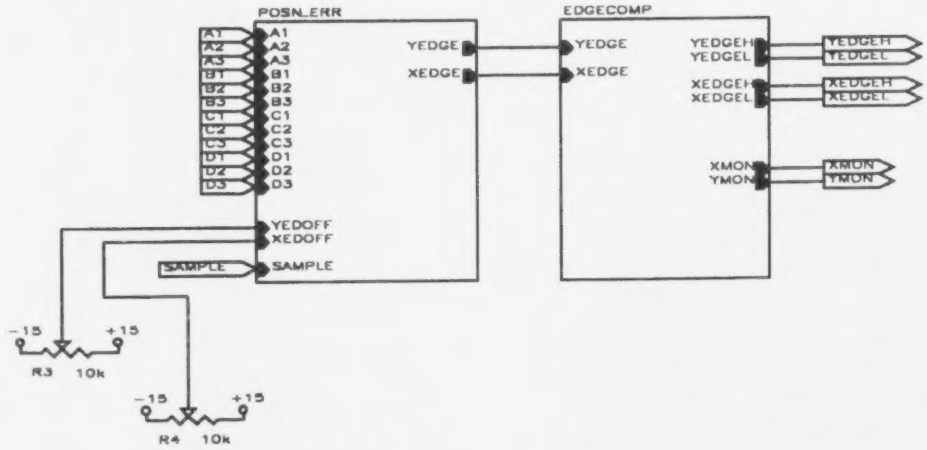


POSITION CONTROLLER CIRCUIT BOARD



SET JUMPERS TO ENABLE POSITION
CONTROL WHEN UNIFORMITY ERROR
IS INSIDE EITHER +/- 2.0 OR
+/- 0.6 VOLT WINDOWS (CONNECT PINS
1/8 AND 2/7 FOR 0.6 WINDOW
OR PINS 3/6 AND 4/5 FOR 2.0 WINDOW)

DESIGNED BY DALE AKITT
Title
POSITION
Size A
Document Number
E:\PIE4\FINAL\POSITION.SCH
REV
Date: April 14, 1992
Sheet of

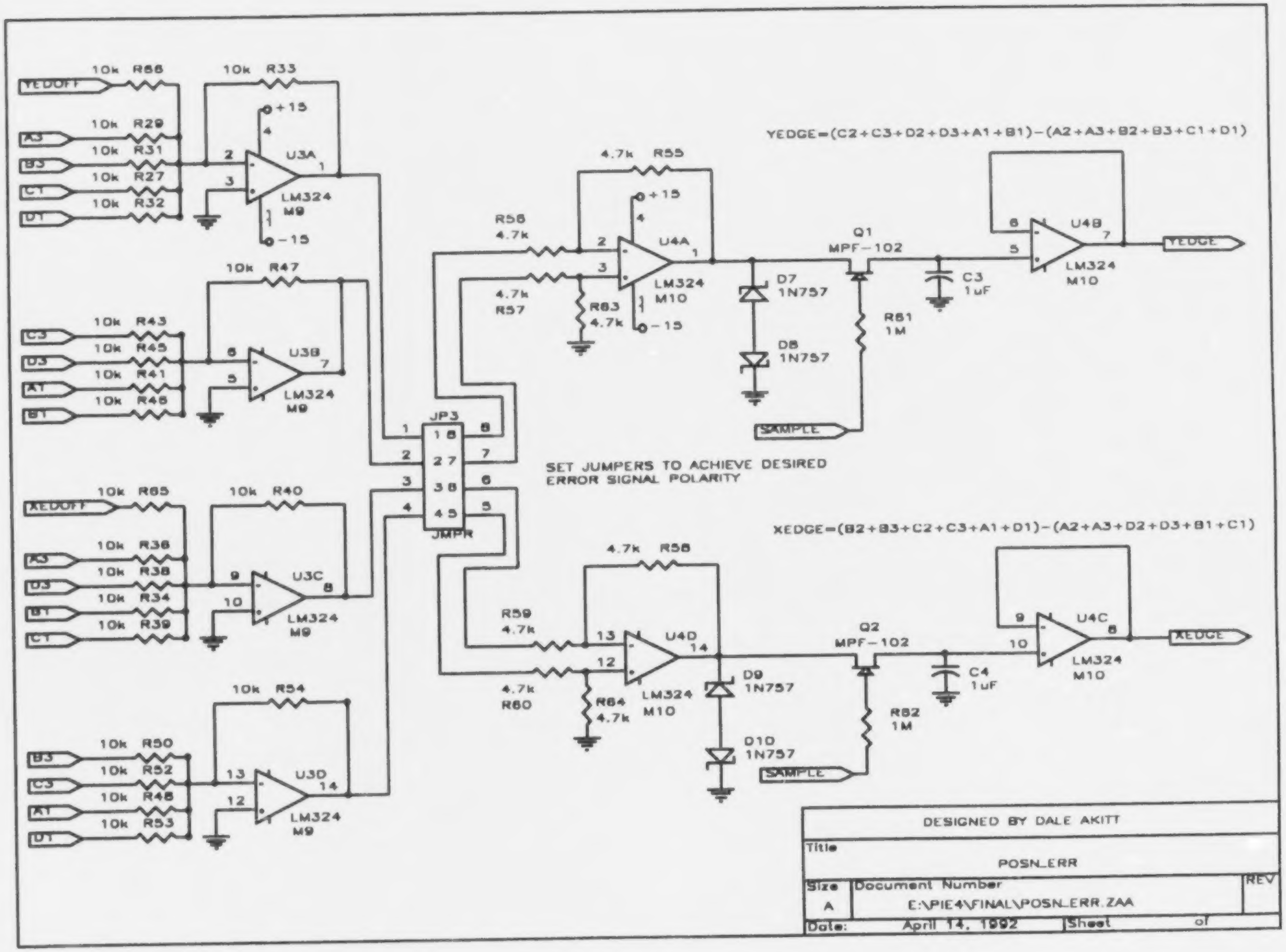


DESIGNED BY DALE AKITT

Title
EDGE_ERR

Size A
Document Number E:\PIE4\FINAL\EDGE_ERR.SCH
REV

Date: April 14, 1992
Sheet of



YEDGE=(C2+C3+D2+D3+A1+B1)-(A2+A3+B2+B3+C1+D1)

SET JUMPERS TO ACHIEVE DESIRED
ERROR SIGNAL POLARITY

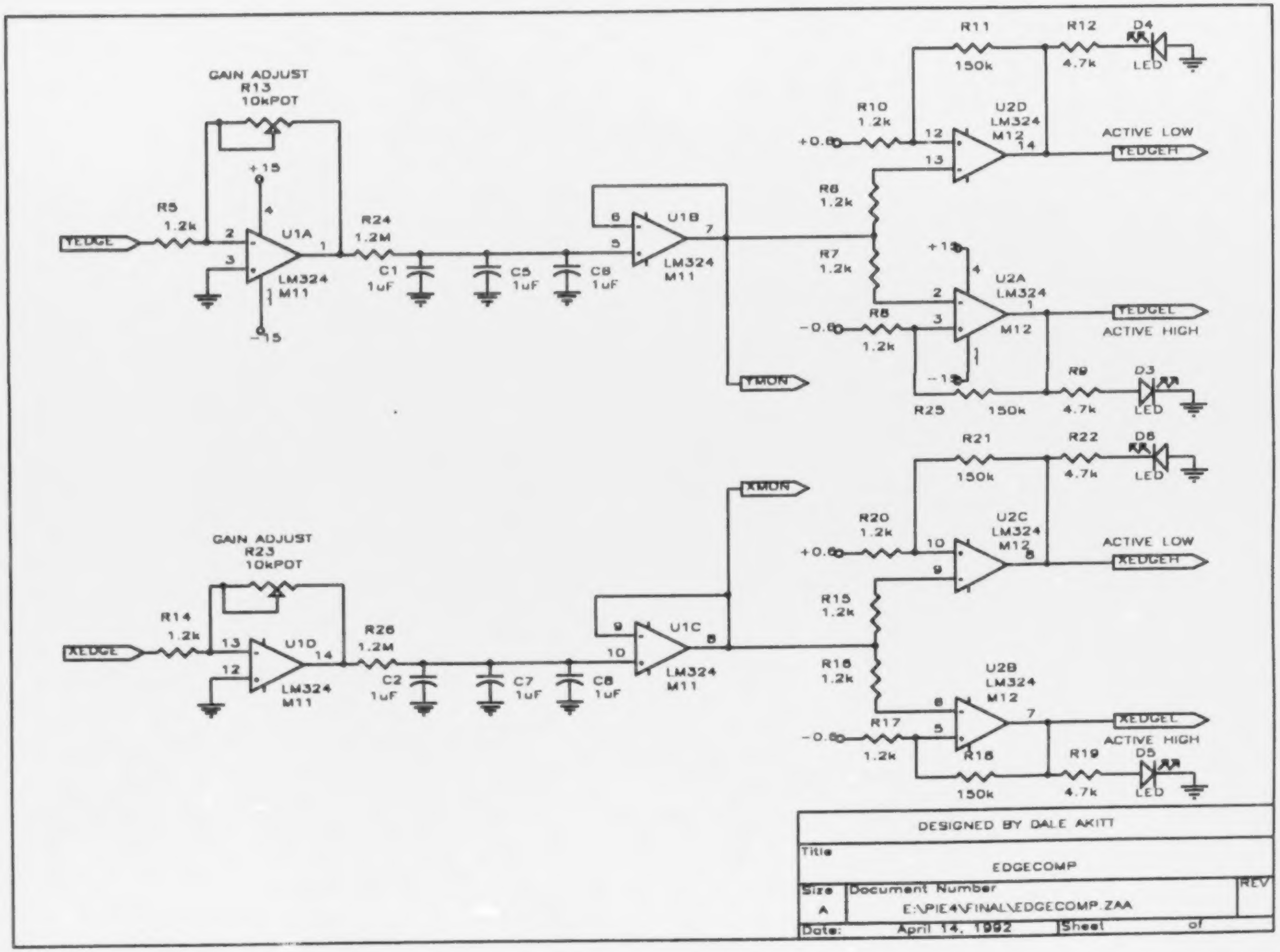
XEDGE=(B2+B3+C2+C3+A1+D1)-(A2+A3+D2+D3+B1+C1)

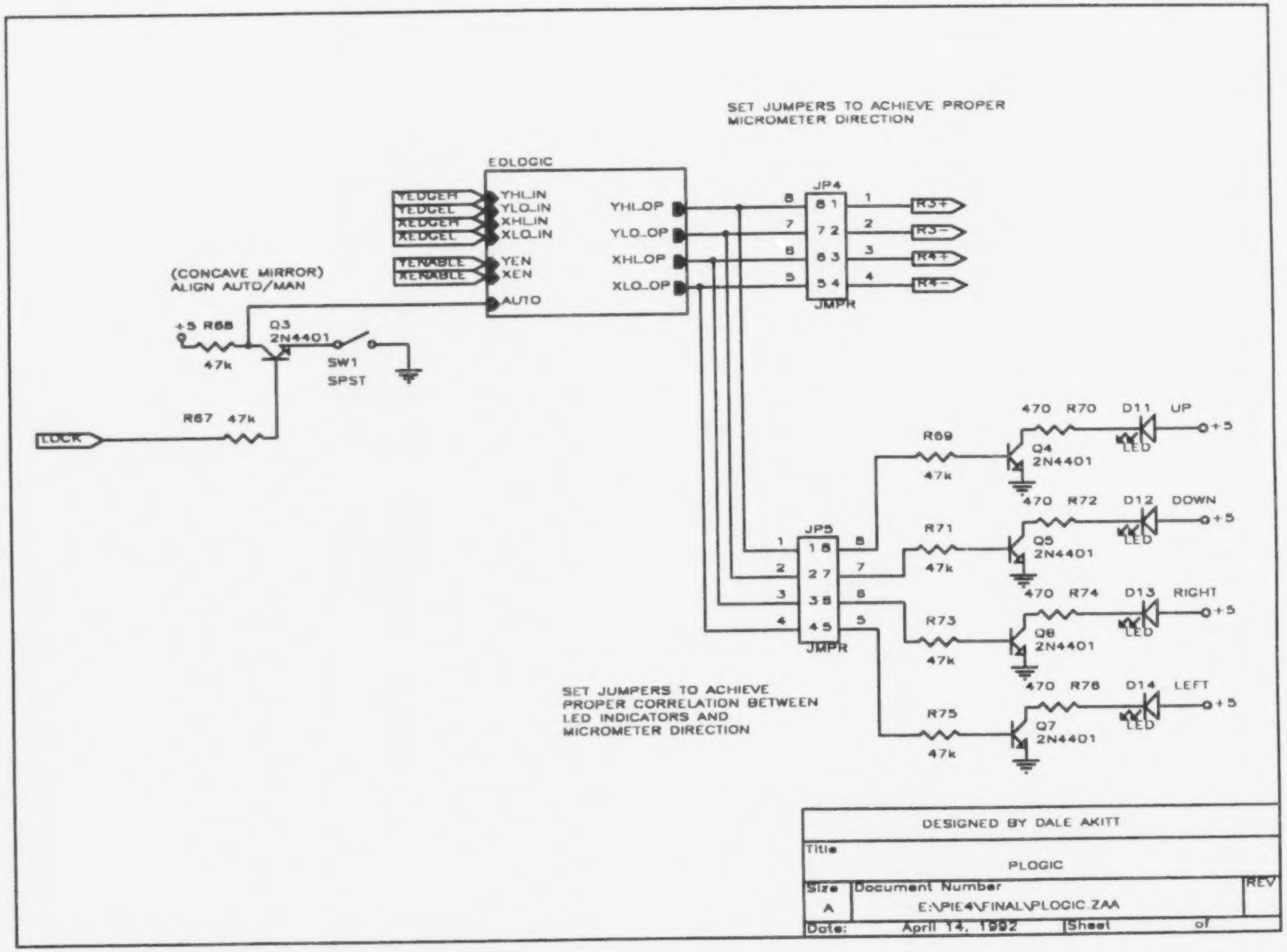
DESIGNED BY DALE AKITT

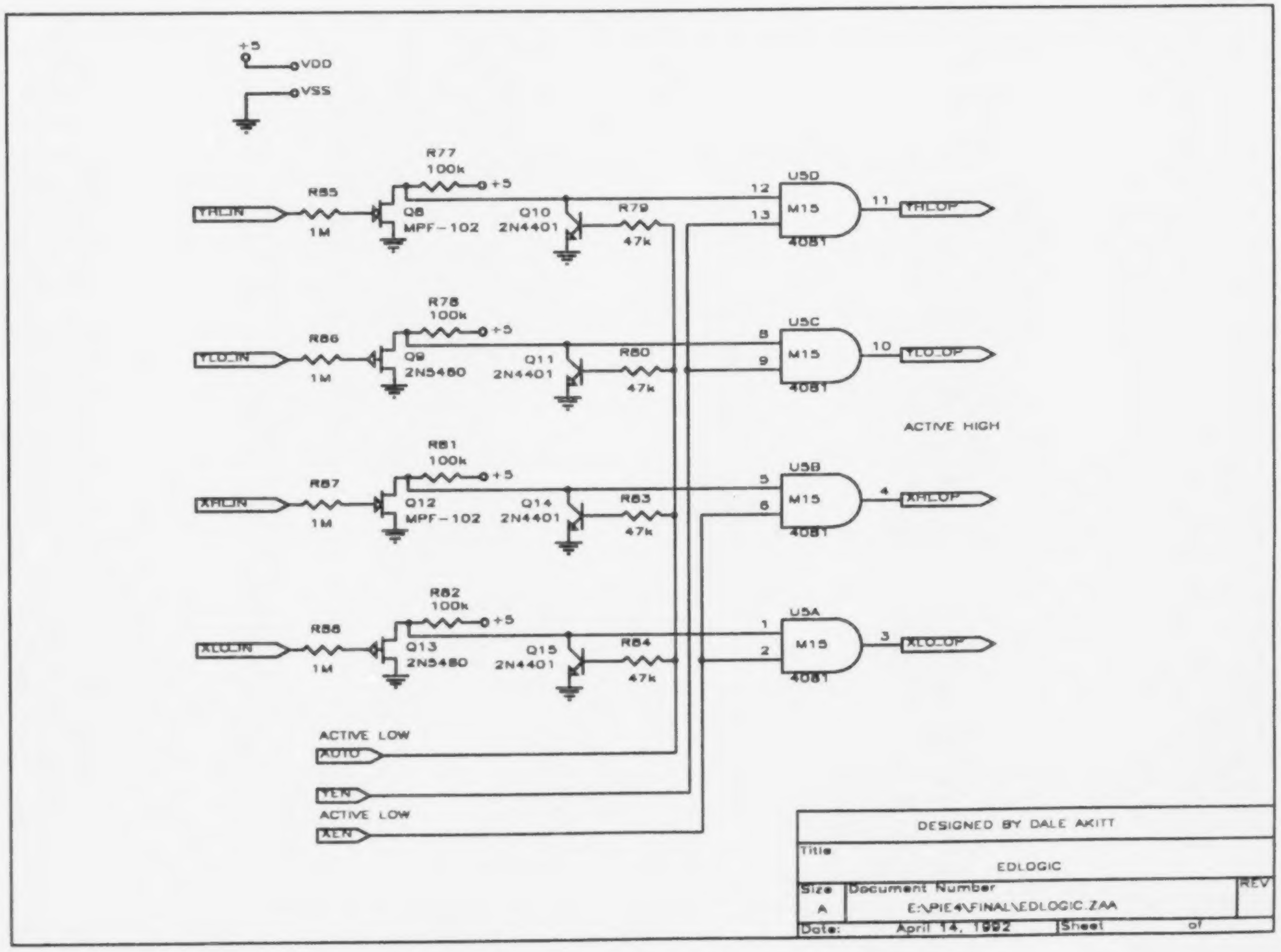
Title
POSN_ERR

Size: A
Document Number: E:\PIE4\FINAL\POSN_ERR.ZAA
REV

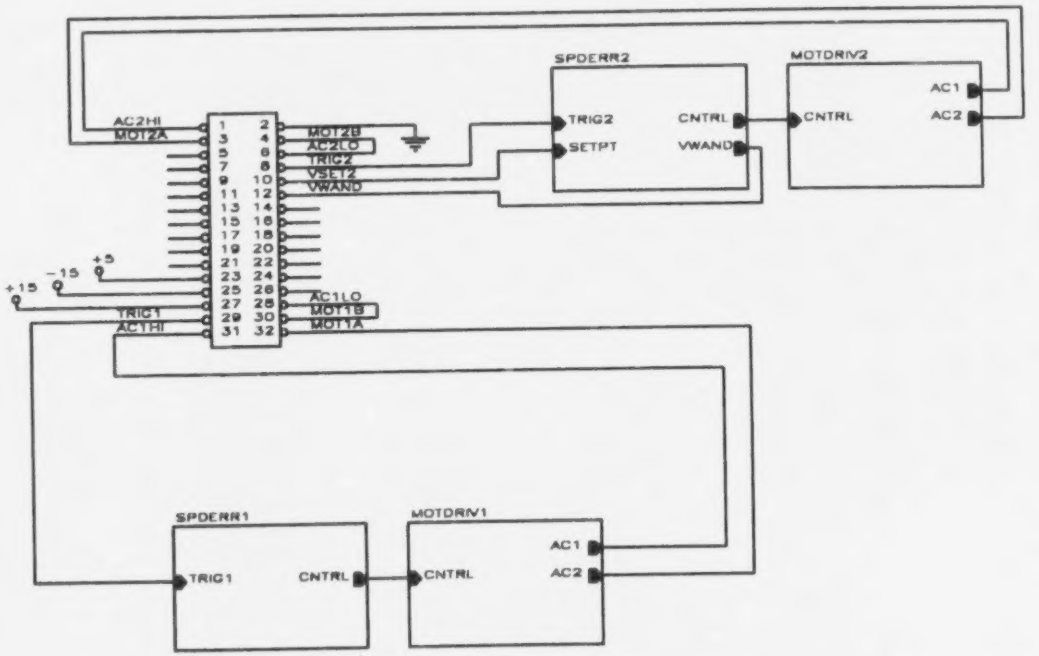
Date: April 14, 1992
Sheet of


GAIN ADJUST
R13
10kPOT
+15
R5
1.2k
YEDGE
U1A
LM324
M11
-15
R24
1.2M
C1
1uF
C5
1uF
C8
1uF
U1B
LM324
M11
YMON
R6
1.2k
R7
1.2k
R8
1.2k
R10
1.2k
+0.8
-0.8
R11
150k
R12
4.7k
D4
LED
U2D
LM324
M12
ACTIVE LOW
YEDGEH
U2A
LM324
M12
+15
-15
YEDGEL
ACTIVE HIGH
R9
4.7k
D3
LED
R25
150k
GAIN ADJUST
R23
10kPOT
R14
1.2k
XEDGE
U1D
LM324
M11
R26
1.2M
C2
1uF
C7
1uF
C8
1uF
U1C
LM324
M11
XMON
R15
1.2k
R16
1.2k
R20
1.2k
R17
1.2k
+0.8
-0.8
R21
150k
R22
4.7k
D6
LED
U2C
LM324
M12
ACTIVE LOW
XEDGEH
U2B
LM324
M12
XEDGEL
ACTIVE HIGH
R18
150k
R19
4.7k
D5
LED
DESIGNED BY DALE AKITT
Title
EDGECOMP
Size
A
Document Number
E:\PIE4\FINAL\EDGECOMP.ZAA
REV
Date:
April 14, 1992
Sheet
of


SET JUMPERS TO ACHIEVE PROPER
MICROMETER DIRECTION
EDLOGIC
YEDGEH
YEDGEL
XEDGEH
XEDGEL
YENABLE
XENABLE
YHL_IN
YLO_IN
XHL_IN
XLO_IN
YEN
XEN
AUTO
YHL_OP
YLO_OP
XHL_OP
XLO_OP
JP4
JMPR
R3+
R3-
R4+
R4-
(CONCAVE MIRROR)
ALIGN AUTO/MAN
+5 R68
47k
Q3
2N4401
SW1
SPST
LOCK
R67 47k
JP5
JMPR
R69
47k
R71
47k
R73
47k
R75
47k
470 R70
470 R72
470 R74
470 R76
Q4
2N4401
Q5
2N4401
Q6
2N4401
Q7
2N4401
D11 UP
D12 DOWN
D13 RIGHT
D14 LEFT
LED
+5
SET JUMPERS TO ACHIEVE
PROPER CORRELATION BETWEEN
LED INDICATORS AND
MICROMETER DIRECTION
DESIGNED BY DALE AKITT
Title
PLOGIC
Size
A
Document Number
E:\PIE4\FINAL\PLOGIC.ZAA
REV
Date:
April 14, 1992
Sheet
of


+5
VDD
VSS
YHI_IN
R85
1M
R77
100k
+5
Q8
MPF-102
Q10
2N4401
R79
47k
U5D
12
13
M15
4081
11
YHI_OP
YLO_IN
R86
1M
R78
100k
+5
Q9
2N5460
Q11
2N4401
R80
47k
U5C
8
9
M15
4081
10
YLO_OP
ACTIVE HIGH
XHI_IN
R87
1M
R81
100k
+5
Q12
MPF-102
Q14
2N4401
R83
47k
U5B
5
6
M15
4081
4
XHI_OP
XLO_IN
R88
1M
R82
100k
+5
Q13
2N5460
Q15
2N4401
R84
47k
U5A
1
2
M15
4081
3
XLO_OP
ACTIVE LOW
AUTO
YEN
ACTIVE LOW
XEN
DESIGNED BY DALE AKITT
Title
EDLOGIC
Size
A
Document Number
E:\PIE4\FINAL\EDLOGIC.ZAA
REV
Date:
April 14, 1992
Sheet
of

WAND SPEED CONTROL CIRCUIT BOARD

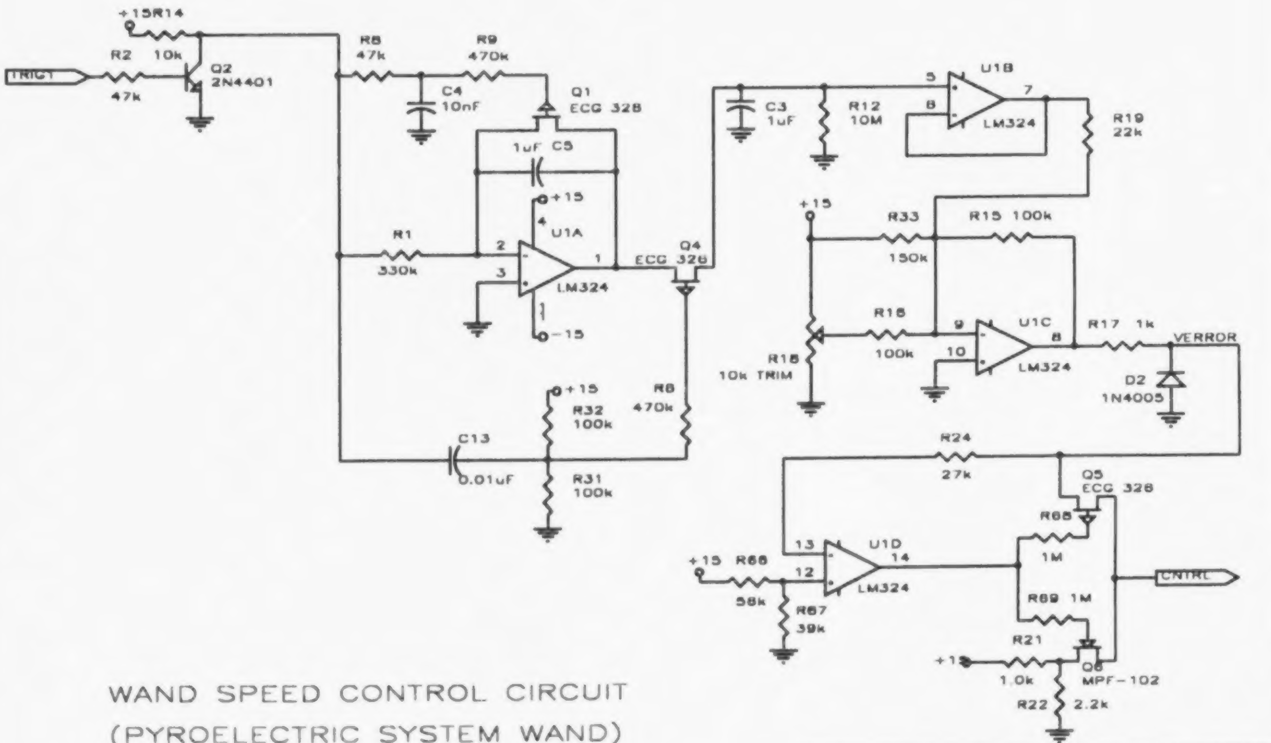


DESIGNED BY DALE AKITT

Title
WANDS

Size A
Document Number E:\PIE4\FINAL\WANDS.SHT
REV

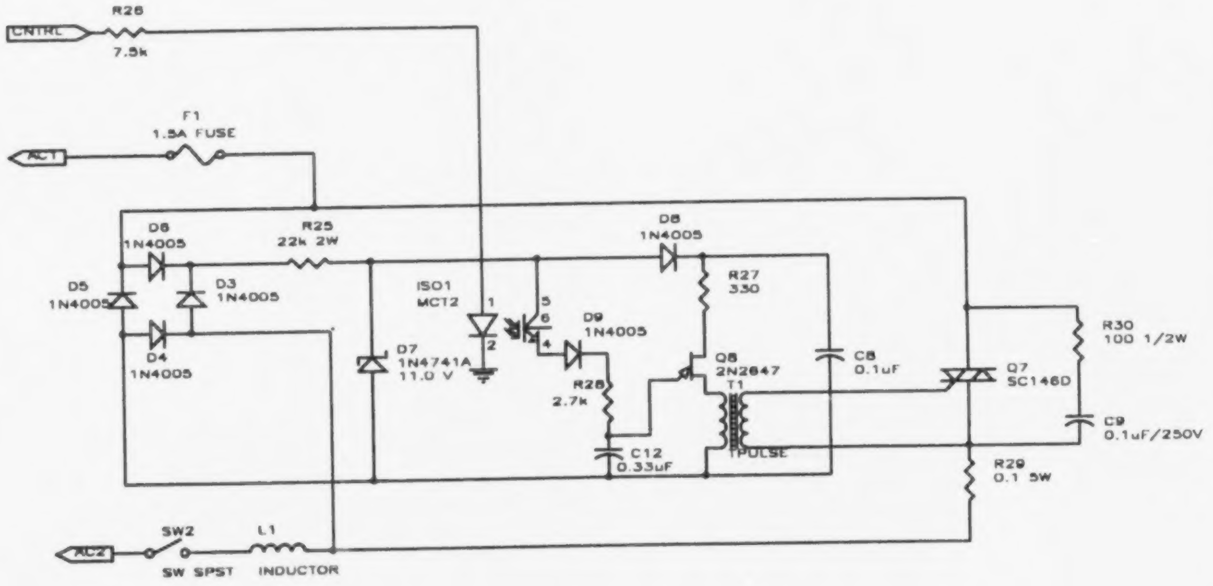
Date: April 14, 1992
Sheet of



WAND SPEED CONTROL CIRCUIT
(PYROELECTRIC SYSTEM WAND)

DESIGNED BY DALE AKITT

Title SPDERR1		
Size A	Document Number E:\PIE4\FINAL\SPDERR1.SHT	REV
Date:	April 14, 1992	Sheet of



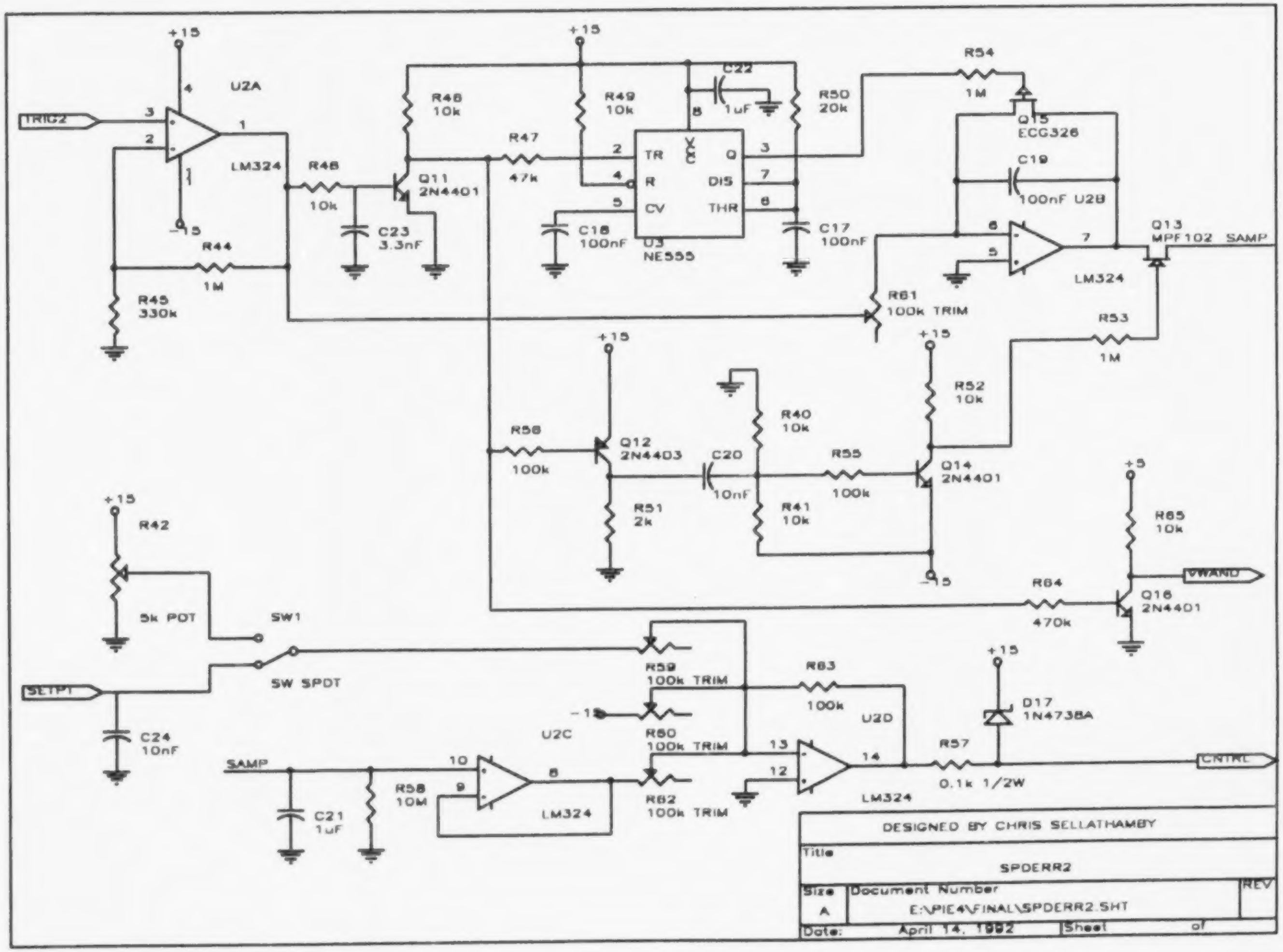
VARIABLE SPEED AC MOTOR DRIVE
(PYROELECTRIC SYSTEM WAND)

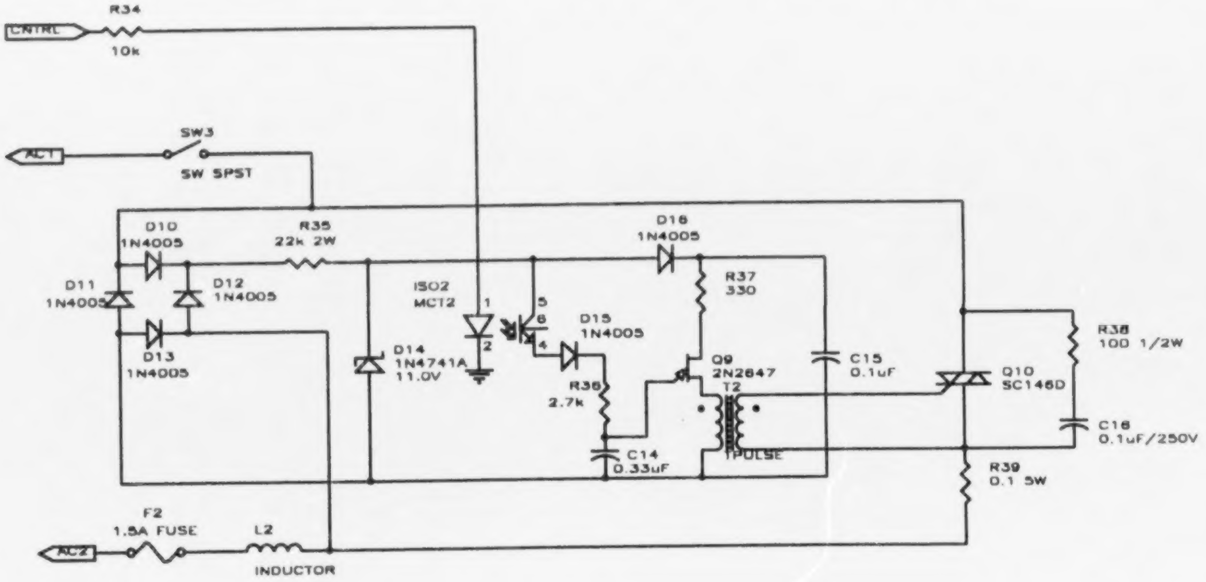
DESIGNED BY DALE AKITT

Title
MOTDRIV1

Size A
Document Number E:\PIE4\FINAL\MOTDRIV1.SHT
REV

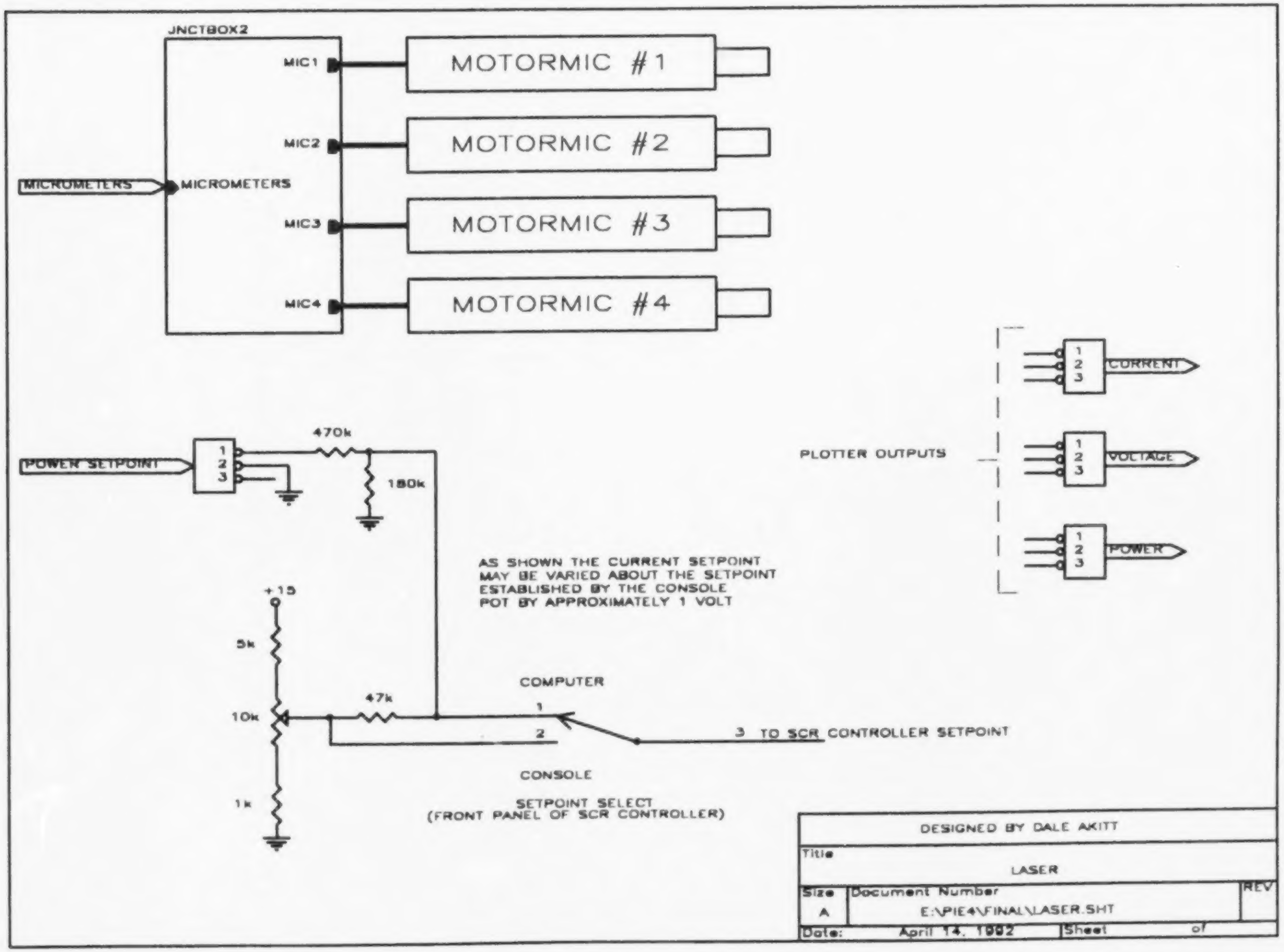
Date: April 14, 1992
Sheet of


TRIG2
+15
U2A
LM324
-15
R44
1M
R45
330k
R48
10k
R46
10k
Q11
2N4401
C23
3.3nF
R47
47k
+15
R49
10k
C22
1uF
R50
20k
TR
R
CV
Q
DIS
THR
VCC
U3
NE555
C18
100nF
C17
100nF
R54
1M
Q15
ECG326
C19
100nF U2B
LM324
Q13
MPF102 SAMP
R61
100k TRIM
R53
1M
+15
R52
10k
Q14
2N4401
R56
100k
Q12
2N4403
C20
10nF
R40
10k
R55
100k
R51
2k
R41
10k
-15
+5
R65
10k
VWAND
R64
470k
Q16
2N4401
+15
R42
5k POT
SW1
SW SPDT
SETPT
C24
10nF
R59
100k TRIM
R60
100k TRIM
R63
100k
U2D
U2C
SAMP
C21
1uF
R58
10M
LM324
R62
100k TRIM
LM324
+15
D17
1N4738A
R57
0.1k 1/2W
CNTRL
DESIGNED BY CHRIS SELLATHAMBY
Title
SPDERR2
Size
A
Document Number
E:\PIE4\FINAL\SPDERR2.SHT
REV
Date:
April 14, 1992
Sheet
of

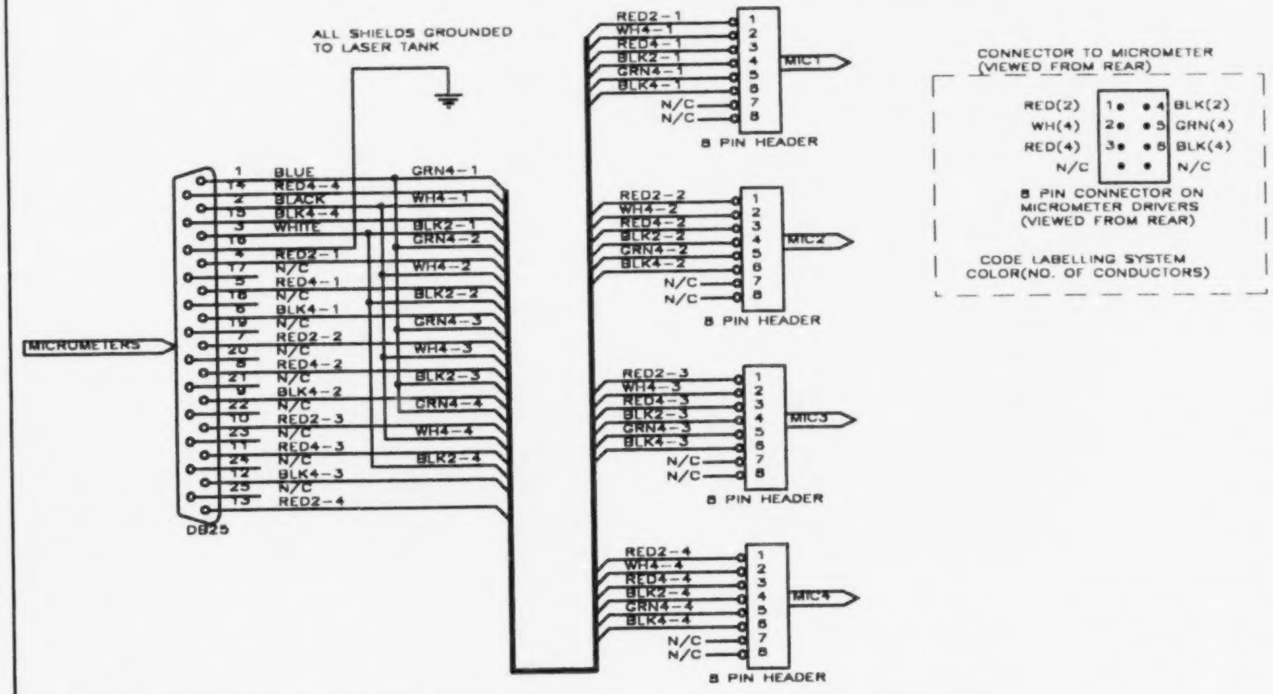


VARIABLE SPEED AC MOTOR DRIVE
(COMPUTER VISION SYSTEM WAND)

DESIGNED BY DALE AKITT			
Title MOTDRIV2			
Size A	Document Number E:\PIE4\FINAL\MOTDRIV2.SHT		REV
Date:	April 14, 1992	Sheet	of


JNCTBOX2
MIC1
MIC2
MIC3
MIC4
MICROMETERS
MICROMETERS
MOTORMIC #1
MOTORMIC #2
MOTORMIC #3
MOTORMIC #4
POWER SETPOINT
1
2
3
470k
180k
AS SHOWN THE CURRENT SETPOINT
MAY BE VARIED ABOUT THE SETPOINT
ESTABLISHED BY THE CONSOLE
POT BY APPROXIMATELY 1 VOLT
+15
5k
10k
1k
47k
COMPUTER
1
2
CONSOLE
3 TO SCR CONTROLLER SETPOINT
SETPOINT SELECT
(FRONT PANEL OF SCR CONTROLLER)
PLOTTER OUTPUTS
CURRENT
VOLTAGE
POWER
DESIGNED BY DALE AKITT
Title
LASER
Size
A
Document Number
E:\PIE4\FINAL\LASER.SHT
REV
Date:
April 14, 1992
Sheet
of

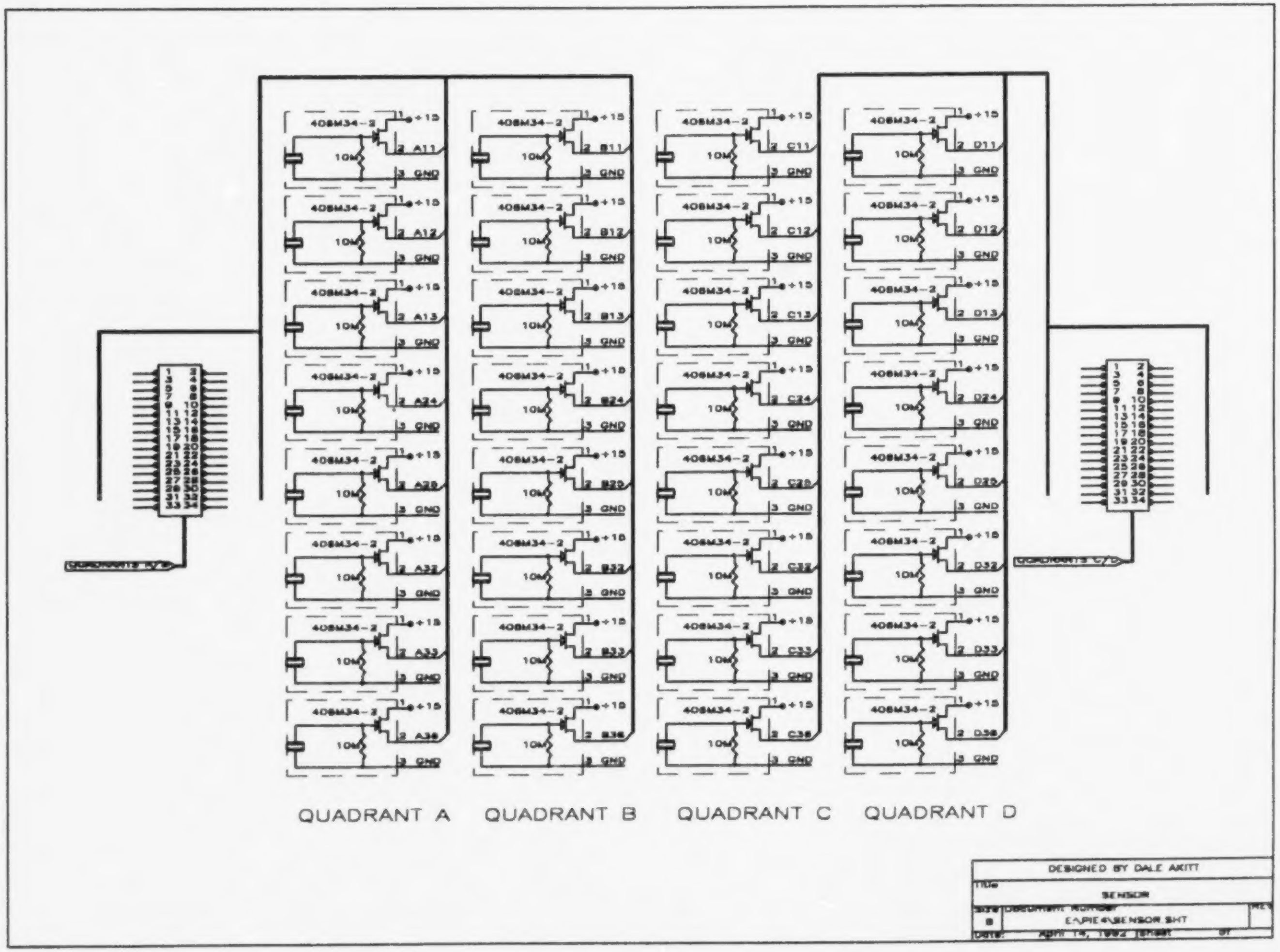
ADAPTER BOX INSIDE LASER (UNDERNEATH MIRRORS)

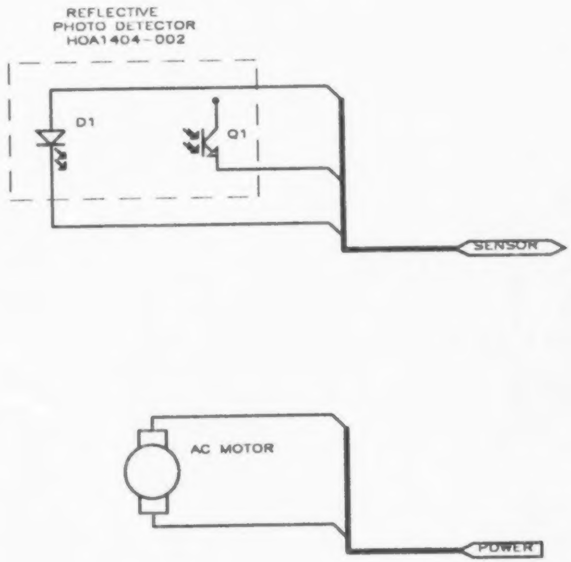


CODING LABELLING SYSTEM
COLOR,NUMBER OF CONDUCTORS - MICROMETER NUMBER

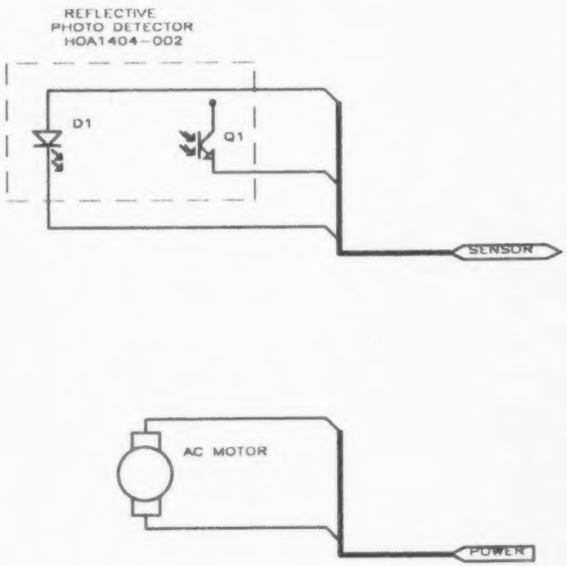
(eg. BLK2-2 indicates black wire in the 2 conductor cable feeding mic #2)

DESIGNED BY DALE AKITT		
Title	JNCTBOX2	
Size A	Document Number E:\PIE4\FINAL\JNCTBOX2.SHT	REV
Date:	April 14, 1992	Sheet of


40BM34-2
10M
+15
GND
A11
A12
A13
A24
A28
A32
A33
A36
B11
B12
B13
B24
B28
B32
B33
B36
C11
C12
C13
C24
C28
C32
C33
C36
D11
D12
D13
D24
D28
D32
D33
D36
QUADRANTS A/B
QUADRANTS C/D
QUADRANT A
QUADRANT B
QUADRANT C
QUADRANT D
DESIGNED BY DALE AKITT
Title
SENSOR
Size
Document Number
B
E:\PIE4\SENSOR.SHT
REV
Date:
April 14, 1992
Sheet
of

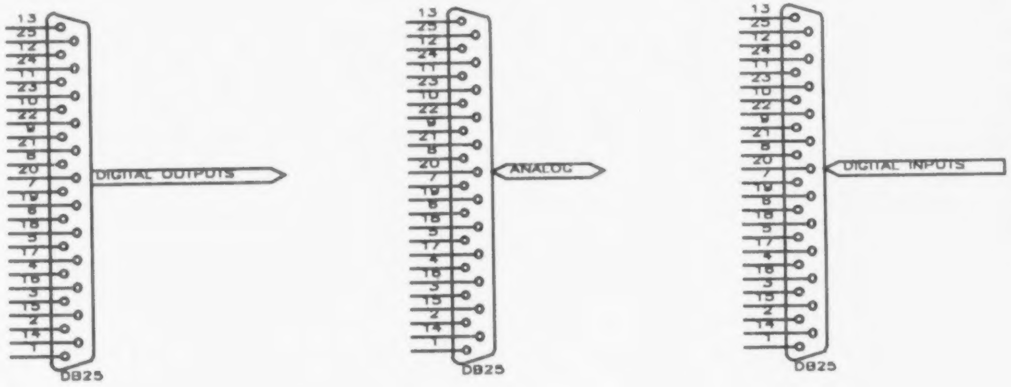


DESIGNED BY DALE AKITT			
Title PYROWAND			
Size A	Document Number E:\PIE4\CARDRACK\PYROWAND.SHT		REV
Date:	April 14, 1992	Sheet of	

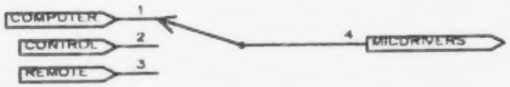


DESIGNED BY DALE AKITT		
Title VISNWAND		
Size A	Document Number E:\PIE4\CARDRACK\VISNWAND.SHT	REV
Date: April 14, 1992	Sheet	of

SEE PROTO–40K MANUAL AND CHRIS SELLATHAMBY'S
THESIS FOR DB25 CONNECTION DETAILS

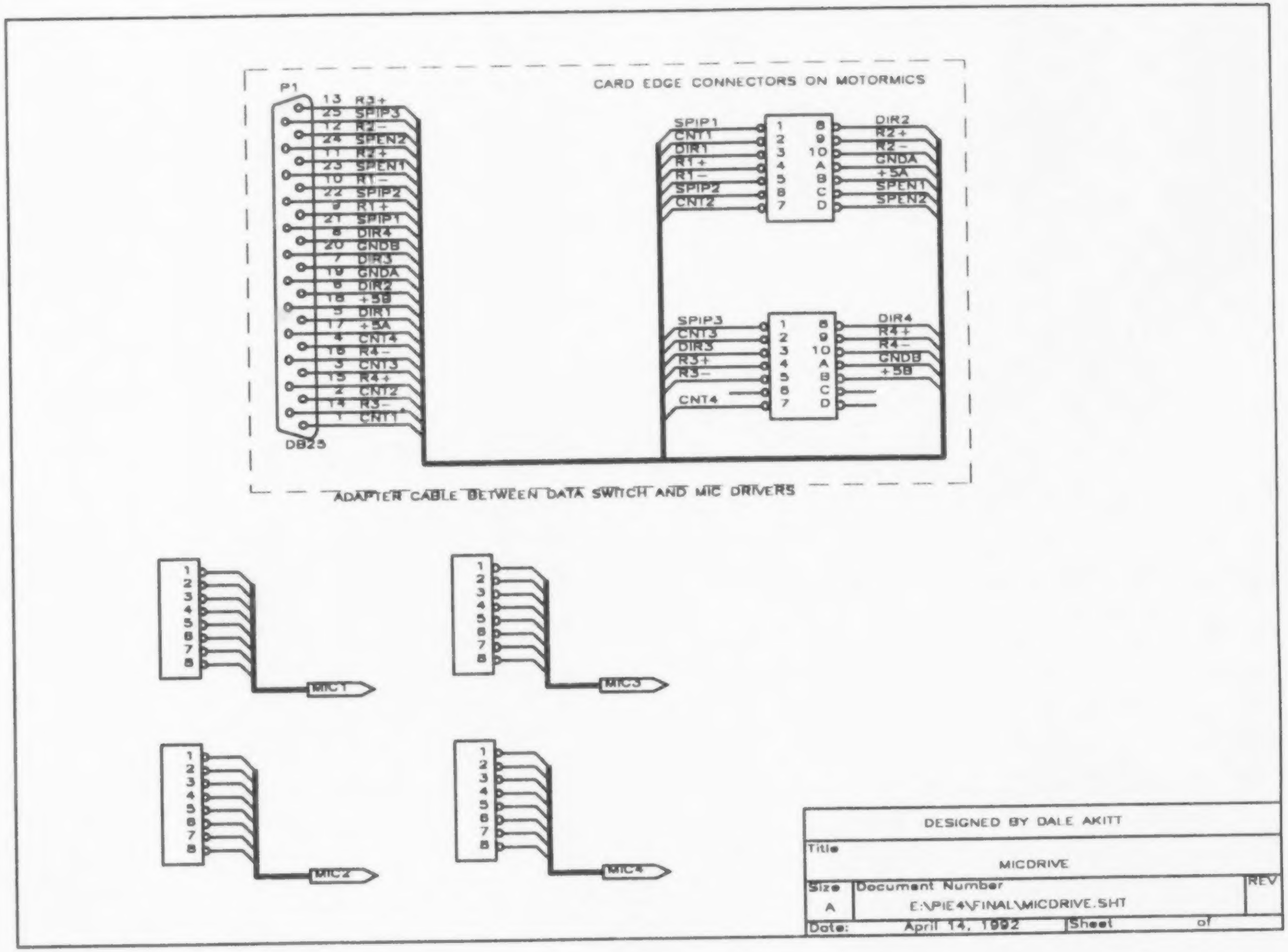


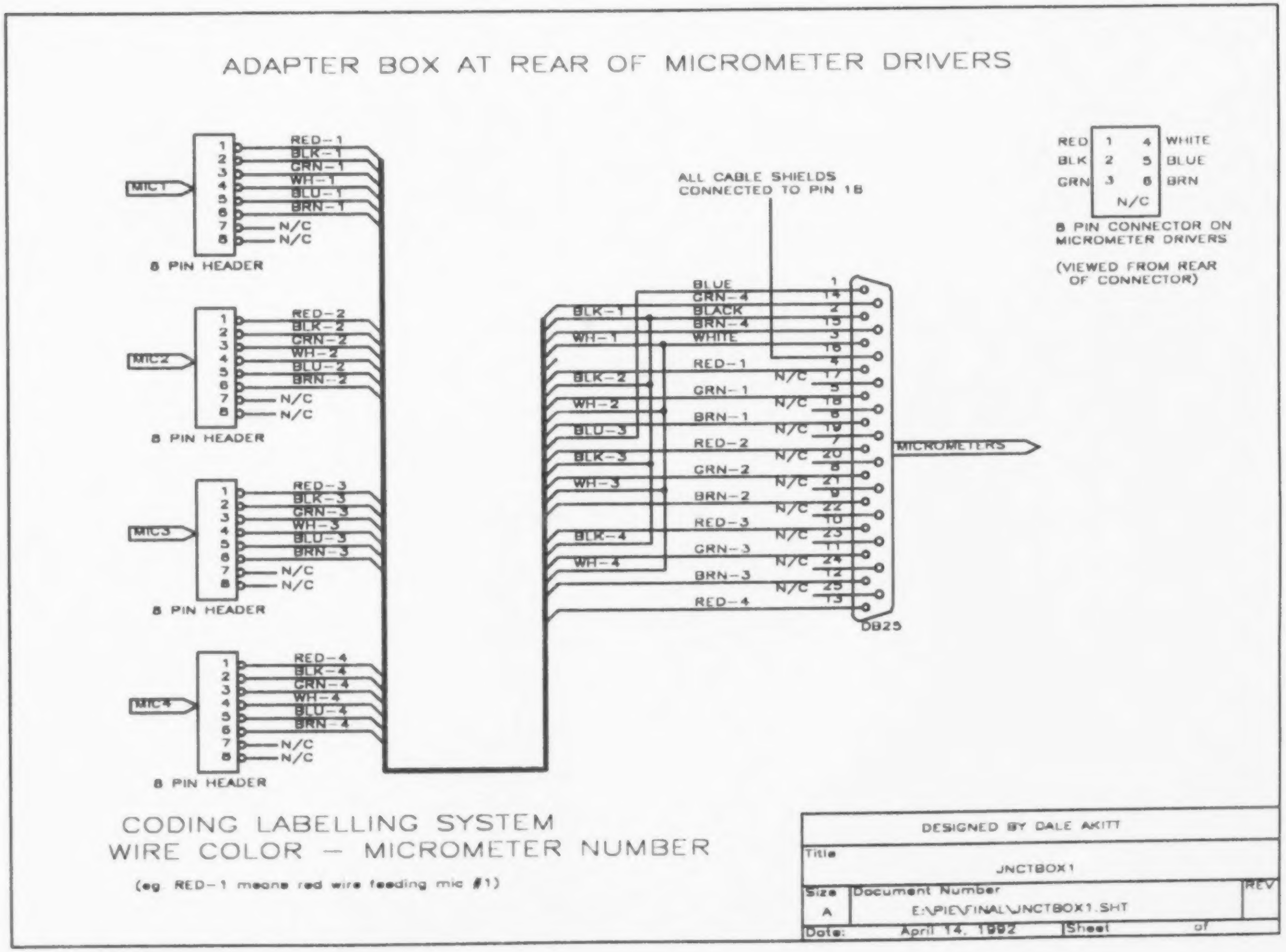
DESIGNED BY DALE AKITT			
Title	AMIGAPC		
Size A	Document Number E:\PIE4\CARDRACK\AMIGAPC.SHT		REV
Date:	April 14, 1992	Sheet	of



MICROMETER CONTROL SELECT

Size A	Document Number		REV
Date: April 11, 1992		Sheet of	


CARD EDGE CONNECTORS ON MOTORMICS
P1
DB25
13 R3+
25 SPIP3
12 R2−
24 SPEN2
11 R2+
23 SPEN1
10 R1−
22 SPIP2
9 R1+
21 SPIP1
8 DIR4
20 GNDB
7 DIR3
19 GNDA
6 DIR2
18 +5B
5 DIR1
17 +5A
4 CNT4
16 R4−
3 CNT3
15 R4+
2 CNT2
14 R3−
1 CNT1
SPIP1
CNT1
DIR1
R1+
R1−
SPIP2
CNT2
1 2 3 4 5 6 7
8 9 10 A B C D
DIR2
R2+
R2−
GNDA
+5A
SPEN1
SPEN2
SPIP3
CNT3
DIR3
R3+
R3−
CNT4
DIR4
R4+
R4−
GNDB
+5B
ADAPTER CABLE BETWEEN DATA SWITCH AND MIC DRIVERS
1 2 3 4 5 6 7 8
MIC1
MIC3
MIC2
MIC4
DESIGNED BY DALE AKITT
Title
MICDRIVE
Size
A
Document Number
E:\PIE4\FINAL\MICDRIVE.SHT
REV
Date:
April 14, 1992
Sheet
of


ADAPTER BOX AT REAR OF MICROMETER DRIVERS
MIC1
MIC2
MIC3
MIC4
1
2
3
4
5
6
7
8
RED-1
BLK-1
GRN-1
WH-1
BLU-1
BRN-1
N/C
N/C
8 PIN HEADER
RED-2
BLK-2
GRN-2
WH-2
BLU-2
BRN-2
RED-3
BLK-3
GRN-3
WH-3
BLU-3
BRN-3
RED-4
BLK-4
GRN-4
WH-4
BLU-4
BRN-4
ALL CABLE SHIELDS CONNECTED TO PIN 16
BLUE
GRN-4
BLACK
BRN-4
WHITE
1
14
2
15
3
16
4
17
5
18
6
19
7
20
8
21
9
22
10
23
11
24
12
25
13
DB25
MICROMETERS
RED
BLK
GRN
WHITE
BLUE
BRN
8 PIN CONNECTOR ON MICROMETER DRIVERS
(VIEWED FROM REAR OF CONNECTOR)
CODING LABELLING SYSTEM
WIRE COLOR – MICROMETER NUMBER
(eg. RED-1 means red wire feeding mic #1)
DESIGNED BY DALE AKITT
Title
JNCTBOX1
Size
A
Document Number
E:\PIE\FINAL\JNCTBOX1.SHT
REV
Date:
April 14, 1992
Sheet
of

APPENDIX IV - SOFTWARE LISTINGS

This appendix contains listings of the final software developed for use on the PIE-4 data-acquisition and control system. These listings contain all of the code used to implement the graphical interface, the PI power controller, and the non-linear alignment controller. The programs were written and compiled on an AMIGA 2000 microcomputer, using an AZTEC C compiler (version 5.2). The following files are listed here:

CONTROL.C	-Main control/data-acquisition program file (pp. 159)
CONTROL.H	-Variable definitions for CONTROL.C (pp. 174)
CONTROL.PAR	-Initialization file which is loaded by CONTROL.C (pp. 178)
SUBS.C	-Support subroutines (pp. 179)
DISPLAY.H	-Variable definitions for the graphical interface (pp. 188)
P40.C	-Subroutines supplied by PROTO-40K manufacturer (pp. 196)
P40.H	-Variable definitions for P40.C (pp. 198)
MAKEFILE	-AZTEC C compiler makefile (pp. 199)

CONTROL.C

```
/***********************************************************************
CONTROL.C-This file contains the main program loop as well as a number of
subroutines.
***********************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <functions.h>
#include <exec/memory.h>
#include <devices/timer.h>
#include <exec/types.h>
#include <intuition/intuition.h>
#include "p40.h"
#include "control.h"
#include "display.h"

/**********************************************************************
SUBROUTINES IN subs.c
**********************************************************************/
void pause(double period);
void move_mic(short mic,short stop_go,short dirn);
void control(void);
void read_data(void);
void reset_timer(void);
void update_timer(void);
void start_timer(void);
void measure_period(void);
void all_stop(void);

/**********************************************************************
SUBROUTINES IN CONTROL.C
**********************************************************************/
void initialize(void);
void clean_up(void);
void line(struct RastPort *RPort,short color,short x0,short y0,
        short x1,short y1);
void set_vars(void);
void store_data(void);
void update_display(void);
void chkmsg(void);
void init_display(void);
void draw_display(void);
void reset_graph(void);
short checkdigs(double time);
void calcerrors(void);
void cntup();
void filter(float *ip,float *op,float *xbuf,float *ybuf,int fn);
void filter_all(void);
void clip(float *ip);
void assign_coefs(float fcutoff,int filtnum,short FiltType);
void read_pars(void);
void read_speeds(void);
void posn_control(void);
void toggleGadget(struct Gadget* gadget);
void SetupDatafile(void);
void PwrControl(void);
void DataSwitch(short PIVOnOff);
void PrintMaxTime(void);
void UpdateDisplays(void);
/*************************************************
  Subroutines in p40.c
*************************************************/
short p40_init(void);
short Open_systimer(void);
short Close_systimer(void);


void main(int argc,char *argv[])
{

initialize();

period = 0.1;
set_vars(); /* initialize global variables used in this program */
init_display();

reset_timer();
start_timer();
update_timer();
```

```
cntrltime = runtime;
YCntrlTime = runtime;
XCntrlTime = runtime;
updatetime = runtime;
PwrTime = runtime;
paramtime = runtime;
DataTimeOffset = runtime;
ldTime0 = runtime;

while(pauseflag)
  chkmsg();

/*  Main loop  */

while(!(quit | no_syncflag))
  {
  while(pauseflag)
    chkmsg();
  read_data();
  update_timer();
  period = runtime - oldtime;
  oldtime = runtime;
  for (i=0;i<last_chan;i++)
    *(volts+i) = *(volts+i) * *(chan_gain+i);
  calcerrors();
  for (i=0;i<last_chan;i++)
    *(volts_avgs+i) += *(volts+i);
  filter_all();
  if ((runtime - PwrTime) > TsPwr)
    {
    PwrControl();
    PwrTime = runtime;
    }
  samp_cntr++;
  if(samp_cntr >= samp_rate)
    {
    samp_cntr = 0;
    for (i=0;i<last_chan;i++)
      *(volts_avgs+i) /= samp_rate;
    store_data();
    for (i=0;i<last_chan;i++)
      *(volts_avgs+i) = 0;
```

```
    update_display();
    }
  control();
  if ((runtime - updatetime) >= 1.0)
    UpdateDisplays();
  chkmsg();
  }

if (no_syncflag)
  printf("\nWARNING - NO SYNC PULSE, CHECK WAND\n");
clean_up();
}

void initialize(void)
{
/*********************************************************************
Configure the Proto-40k and map its registers.  See p40.c for details.
*********************************************************************/
if(p40_init()==0)
   {
   printf("Couldn't configure the Proto-40k\n");
   exit(0);
   }
*p40_modeselect = 0;
*p40_gain = 0;

/*****************************************************************
   Allocate memory for analog data (raw and filtered)
*****************************************************************/
volts = (float*)calloc((size_t)last_chan, (size_t)sizeof(float));
volts_avgs = (float*)calloc((size_t)last_chan, (size_t)sizeof(float));
filtvolts = (float*)calloc((size_t)last_chan,(size_t)sizeof(float));
/*********************************************************************
   set up a timer for time keeping during the run
*********************************************************************/
Open_systimer();
reset_timer();

DtoA = 255/2;
*p40_da1 = DtoA;
Upwr = 7.5;
}
```

```
void clean_up(void)
{
all_stop();
CloseWindow(DispWind);
CloseScreen(Screen1);
Close_systimer();
free(volts);
free(volts_avgs);
free(filtvolts);
if (file_data)
  fclose(datafp);
}

void line(struct RastPort *RPort,short color,short x0,short y0,
   short x1,short y1)
{
Move(RPort,(int)x0,(int)y0);
SetAPen(RPort,(int)color);
Draw(RPort,(int)x1,(int)y1);
}

void set_vars(void)
{
UPMAX = 15.0;    /* maximum allowed analog output (d/a 1) */
filtflag = FALSE;
sumflag = FALSE;
pauseflag = TRUE;
no_syncflag = FALSE;
first_time = TRUE;
quit = FALSE;
pivflag = TRUE;
samp_rate = 1;
pt_spacing = 1;
samp_cntr = 0;
yscale = (lowy - highy)/20.0; /* scale volts to pixels */
read_pars();  /* read controller parameters (pi4:test/control.pars) */

/* initial settings for position control sampling periods*/
Tposx = 1.0;
Tposy = 1.0;
```

```
read_speeds(); /* read the micrometer speeds (previously recorded in
          micspeeds.pars by running "setspeed") */
/*********************************************************************
Choose the analog channels which are to be recorded in the datafile
*********************************************************************/
selected_chans[0] = TRUE;
selected_chans[1] = TRUE;
selected_chans[2] = TRUE;
selected_chans[3] = TRUE;
selected_chans[4] = TRUE;
selected_chans[5] = TRUE;
selected_chans[6] = TRUE;
selected_chans[7] = TRUE;
selected_chans[8] = TRUE;
selected_chans[9] = TRUE;
selected_chans[10] = TRUE;
selected_chans[11] = TRUE;
selected_chans[12] = TRUE;
selected_chans[13] = TRUE;
selected_chans[14] = TRUE;
selected_chans[15] = TRUE;
/*********************************************************************
  Initialize such that no channels are plotted initially, they are
  then selected or deselected using the gadgets on the display screen
*********************************************************************/
graph_chans[0] = TRUE;
graph_chans[1] = FALSE;
graph_chans[2] = FALSE;
graph_chans[3] = FALSE;
graph_chans[4] = FALSE;
graph_chans[5] = FALSE;
graph_chans[6] = FALSE;
graph_chans[7] = FALSE;
graph_chans[8] = FALSE;
graph_chans[9] = FALSE;
graph_chans[10] = FALSE;
graph_chans[11] = FALSE;
graph_chans[12] = FALSE;
graph_chans[13] = FALSE;
graph_chans[14] = FALSE;
graph_chans[15] = FALSE;
```

```
/*  Set the analog channel gains
    (mainly to invert the negative sensor signals)   */

chan_gain[0] = 1.0;
chan_gain[1] = 1.0;
chan_gain[2] = 1.0;
chan_gain[3] = 1.0;
chan_gain[4] = -1.0;
chan_gain[5] = -1.0;
chan_gain[6] = -1.0;
chan_gain[7] = -1.0;
chan_gain[8] = -1.0;
chan_gain[9] = -1.0;
chan_gain[10] = -1.0;
chan_gain[11] = -1.0;
chan_gain[12] = -1.0;
chan_gain[13] = -1.0;
chan_gain[14] = -1.0;
chan_gain[15] = -1.0;
}

/*****************************************************
store_data saves the data to the ram
file IFF the DATA gadget is selected
*****************************************************/

void store_data(void)

{
short i;
if(file_data)
  {
  if (IdFlag)
    fprintf(datafp,"%8.3f %2d %5.1f  ",runtime-DataTimeOffset,
      MicIp,mic_cntr[CurMic]/10.0);
  else
    {
    fprintf(datafp,"%8.3f %1d %1d %1d ",runtime-
DataTimeOffset,M1auto,M2auto,PwrAuto);
    fprintf(datafp,"%5.2f %5.2f",PwrSetpt,Upwr);
    }
  if (pivflag)
```

```
    {
    fprintf(datafp,"%5.2f %5.2f %5.2f ",power,current,voltage);
    for (i=3;i<last_chan;i++)
      if(*(selected_chans+i)==1)
        fprintf(datafp,"%5.2f ",*(volts_avgs+i));
    }
  else
    {
    for (i=0;i<last_chan;i++)
      if(*(selected_chans+i)==1)
        fprintf(datafp,"%5.2f ",*(volts_avgs+i));
    }
  fprintf(datafp,"\r\n");
  }
}

void read_pars(void)
{
char parname[20];
char parvalue[10];
char parval1[10],parval2[10],parval3[10],parval4[10];
char comment[100];
int input;

if ((paramfp = fopen("control.pars","r")) == 0)
  {
  printf("Can't open parameter file! (CONTROL.PARS)\n");
  quit = TRUE;
  }
if (!quit)
  {
  input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
  PwrSetpt = atof(parvalue);
  input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
  IdleSetpt = atof(parvalue);
  input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
  KcPwr = atof(parvalue);
  input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
  TiPwr = atof(parvalue);
  input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
  TsPwr = atof(parvalue);
  input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
```

```
samp_rate = atof(parvalue);
input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
TposMax = atof(parvalue);
input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
TposMin = atof(parvalue);
input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
PosnPulseLen = atof(parvalue);
input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
Kcyp = atof(parvalue);
input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
Kcxp = atof(parvalue);
input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
Kcyu = atof(parvalue);
input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
Kcxu = atof(parvalue);
input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
yoffsetp = atof(parvalue);
input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
xoffsetp = atof(parvalue);
input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
yoffsetu = atof(parvalue);
input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
xoffsetu = atof(parvalue);
input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
fcposn = atof(parvalue);
input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
fcunif = atof(parvalue);
input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
fcchan = atof(parvalue);
input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
fcpwr = atof(parvalue);
input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
CurFiltType = atof(parvalue);
input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
PwrFactor = atof(parvalue);
input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
CurFactor = atof(parvalue);
input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
VolFactor = atof(parvalue);
input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
PwrOffset = atof(parvalue);
input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
```

```
  CurOffset = atof(parvalue);
  input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
  VolOffset = atof(parvalue);
  input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
  pyrofactor = atof(parvalue);
  input = fscanf(paramfp,"%s %s %[^\n]",parname,parvalue,comment);
  pyrooffset = atof(parvalue);

  input = fscanf(paramfp,"%s %s %s %s %s
%[^\n]",parname,parval1,parval2,parval3,parval4,comment);
  MicBacklash[1] = atof(parval1);
  MicBacklash[2] = atof(parval2);
  MicBacklash[3] = atof(parval3);
  MicBacklash[4] = atof(parval4);

  fclose(paramfp);
  period = 0.1;
  assign_coefs(fcchan,CHANFILTER,CurFiltType);
  assign_coefs(fcposn,POSNFILTER,CurFiltType);
  assign_coefs(fcunif,UNIFFILTER,CurFiltType);
  assign_coefs(fcpwr,POWERFILTER,CurFiltType);
  if (ScreenFlag)
   {
   if (samp_rate != OldSampRate)
     {
     PrintMaxTime();
     OldSampRate = samp_rate;
     }
   toggleGadget(&PARSGadget);
   }
  }
}


void update_display(void)
{
short i;

/* Clip all values so they fall inside +/- 10 */

if(filtflag)
```

```
  for (i=0;i<16;i++)
    clip(filtvolts+i);
else
  for (i=0;i<16;i++)
    clip(volts+i);
clip(&sumtotal);
clip(&YUnifErr);
clip(&XUnifErr);
clip(&YPosErr);
clip(&XPosErr);
/*********************************************************************
   Setup for the beginning of the plot
*********************************************************************/
if (graphx == 0)
  {
  old_x = initx + 1;
  graphx = initx + pt_spacing;
  if (filtflag == TRUE)
    for (i=0;i<16;i++)
      *(old_y+i) = midpt - floor(*(filtvolts+i)*yscale);
  else
    for (i=0;i<16;i++)
      *(old_y+i) = midpt - floor(*(volts+i)*yscale);
  oldtotal = midpt - floor(sumtotal*yscale);
  OldXUnifErr = midpt - floor(XUnifErr*yscale);
  OldYUnifErr = midpt - floor(YUnifErr*yscale);
  OldXPosErr = midpt - floor(XPosErr*yscale);
  OldYPosErr = midpt - floor(YPosErr*yscale);
  OldPwrSetpt = midpt - floor(PwrSetpt * yscale);
  OldUpwrPlot = midpt - floor(UpwrPlot * yscale);
  }
/*********************************************************************
   Scroll if the plot has reached the right
   side of the plotting display
*********************************************************************/
if (graphx >= finalx)
  {
  old_x = old_x - (scrollpt - initx - 1);
  graphx = old_x + pt_spacing;
  if (CheckSpeedFlag)
    {
    Move(MyRPort,(finalx - initx)/2 + initx - 50,highy+10);
    SetAPen(MyRPort,(int)graph_back_color);
    Text(MyRPort,"SET WAND SPEED TO 100 ms",24);
    }
  reset_graph();
  if (CheckSpeedFlag)
    {
    Move(MyRPort,(finalx - initx)/2 + initx - 50,highy+10);
    SetAPen(MyRPort,(int)graph_color);
    Text(MyRPort,"SET WAND SPEED TO 100 ms",24);
    }
  }
for (i=0;i<16;i++)
  {
  if(*(graph_chans+i) == TRUE)
    {
    if (filtflag)
      graphy = midpt - floor(*(filtvolts+i)*yscale);
    else
      graphy = midpt - floor(*(volts+i)*yscale);
    line(MyRPort,*(chan_cols+i),old_x,*(old_y+i),graphx,graphy);
    *(old_y+i) = graphy;
    }
  }
if (sumflag)
  {
  graphy = midpt - floor(sumtotal*yscale);
  line(MyRPort,white,old_x,oldtotal,graphx,graphy);
  oldtotal = graphy;
  }
if (UNXFlag)
  {
  graphy = midpt - floor(XUnifErr*yscale);
  line(MyRPort,UNXCol,old_x,OldXUnifErr,graphx,graphy);
  OldXUnifErr = graphy;
  }
if (UNYFlag)
  {
  graphy = midpt - floor(YUnifErr*yscale);
  line(MyRPort,UNYCol,old_x,OldYUnifErr,graphx,graphy);
  OldYUnifErr = graphy;
  }
if (POXFlag)
```

```
 {
 graphy = midpt - floor(XPosErr*yscale);
 line(MyRPort,POXCol,old_x,OldXPosErr,graphx,graphy);
 OldXPosErr = graphy;
 }
if (POYFlag)
 {
 graphy = midpt - floor(YPosErr*yscale);
 line(MyRPort,POYCol,old_x,OldYPosErr,graphx,graphy);
 OldYPosErr = graphy;
 }
if (PspFlag)
 {
 graphy = midpt - floor(PwrSetpt*yscale);
 line(MyRPort,PspCol,old_x,OldPwrSetpt,graphx,graphy);
 OldPwrSetpt = graphy;
 }
if (UpwrFlag)
 {
 graphy = midpt - floor(UpwrPlot*yscale);
 line(MyRPort,UpwrCol,old_x,OldUpwrPlot,graphx,graphy);
 OldUpwrPlot = graphy;
 }
old_x = graphx;
graphx += pt_spacing;
}


/*****************************************************************
   chkmsg() monitors the display panel gadgets and
   takes appropriate action when various gadgets are
   selected or deselected
*****************************************************************/
void chkmsg(void)
{
ULONG MICode;
ULONG MIClass;
APTR MIAddress;

while(message=(struct IntuiMessage *)GetMsg(DispWind->UserPort))
 {
 MICode = message->Code;
```

```
MIClass = message->Class;
MIAddress = message->IAddress;
ReplyMsg((struct Message *) message);
switch(MIClass)
 {
 case CLOSEWINDOW:
  while (message = (struct IntuiMessage *) GetMsg(DispWind->UserPort))
    ReplyMsg((struct Message *)message);
  clean_up();
  printf("\n");
  exit(0);
  break;
 case GADGETDOWN:
  for (i=0;i<16;i++)
   if(MIAddress == (APTR)&chanGadget[i])
     graph_chans[i] = !graph_chans[i];
  if (MIAddress == (APTR)&pauseGadget)
   {
   pauseflag = !pauseflag;
   if(!pauseflag)
     start_timer();
   else
     {
     stop_timer();
     all_stop();
     }
   }
  else if (MIAddress == (APTR)&PARSGadget)
   read_pars();
  else if (MIAddress == (APTR)&DataGadget)
   {
   file_data = !file_data;
   if (!file_data)
     fclose(datafp);
   if (file_data)
     SetupDatafile();
   }
  else if (MIAddress == (APTR)&UNXGadget)
   UNXFlag = !UNXFlag;
  else if (MIAddress == (APTR)&UNYGadget)
   UNYFlag = !UNYFlag;
  else if (MIAddress == (APTR)&POXGadget)
  POXFlag = !POXFlag;
 else if (MIAddress == (APTR)&POYGadget)
  POYFlag = !POYFlag;
 else if (MIAddress == (APTR)&PspGadget)
  PspFlag = !PspFlag;
 else if (MIAddress == (APTR)&UpwrGadget)
  UpwrFlag = !UpwrFlag;
 else if(MIAddress == (APTR)&M1AUTOGadget)
  {
  M1auto = !M1auto;
  if (!M1auto)
   {
   all_stop();
   if (M2auto)
    {
    M2auto = FALSE;
    toggleGadget(&M2AUTOGadget);
    }
   }
  }
 else if(MIAddress == (APTR)&M2AUTOGadget)
  {
  M2auto = !M2auto;
  if (M2auto) cntrltime = runtime;
  if (!M2auto)
   {
   all_stop();
   movingyposn = FALSE;
   movingxposn = FALSE;
   }
  else if (!M1auto)
   {
   toggleGadget(&M1AUTOGadget);
   M1auto = !M1auto;
   }
  }
 else if(MIAddress == (APTR)&SumGadget)
  sumflag = !sumflag;
 else if(MIAddress == (APTR)&FiltGadget)
  {
  filtflag = !filtflag;
  if (filtflag)
```

```
    fprintf(datafp,"\n");
   }
  else if(MIAddress == (APTR)&PAUTOGadget)
   {
   if (pivflag)
    {
    PwrAuto = !PwrAuto;
    if (!PwrAuto)
     IdleSetpt = Upwr;
    }
   else
    toggleGadget(&PAUTOGadget);
   }
  else if(MIAddress == (APTR)&PIVGadget)
   {
   pivflag = !pivflag;
   DataSwitch(pivflag);
   }
 default:
  break;
 }
}
}

void init_display(void)
{

/*************************************************************************
   Open a screen and window to use for displaying the data
   collected from the pyroelectric sensor and alignment
   control system
*************************************************************************/

IntuitionBase = (struct IntuitionBase*)
OpenLibrary("intuition.library",INTUITION_REV);
if(IntuitionBase == NULL)
 exit(FALSE);
GfxBase = (struct GfxBase*)OpenLibrary("graphics.library",GRAPHICS_REV);
if(GfxBase == NULL)
 exit(FALSE);
if((Screen1 = (struct Screen*)OpenScreen(&NewScreen)) == NULL)
 exit(FALSE);
LoadRGB4(&Screen1->ViewPort,colcmap,16);
```

```
/************************************************************************
   Initialize the NewWindow structure to use
   for opening the display window
************************************************************************/
NewWindow.LeftEdge = 0;
NewWindow.TopEdge = 10;
NewWindow.Width = width;
NewWindow.Height = height-10;
NewWindow.DetailPen = darkblue;
NewWindow.BlockPen = white;
NewWindow.Title = NULL;
NewWindow.Flags = WINDOWCLOSE|SMART_REFRESH|ACTIVATE
  |WINDOWDEPTH|GIMMEZEROZERO|RMBTRAP;
NewWindow.IDCMPFlags = CLOSEWINDOW|GADGETDOWN;
NewWindow.Type = CUSTOMSCREEN;
NewWindow.FirstGadget = &chanGadget[0];
NewWindow.CheckMark = NULL;
NewWindow.Screen = Screen1;
NewWindow.BitMap = NULL;

if ((DispWind = (struct Window*)OpenWindow(&NewWindow)) == NULL)
  quit = 1;
/************************************************************************
   Initialize the display
************************************************************************/
MyRPort = DispWind -> RPort;
SetDrMd(MyRPort,JAM2);
SetBPen(MyRPort,(int)graph_back_color);
/*********************************************************************
   Initialize the channel selection gadgets
*********************************************************************/
for (i=15;i>=0;i--)
 {
  /* Text structures first */

  chanText[i].FrontPen = chan_cols[i];
  chanText[i].BackPen = blue;
  chanText[i].DrawMode = JAM1;
  if (i < 10)
    chanText[i].LeftEdge = 5;
  else
    chanText[i].LeftEdge = 2;
  chanText[i].TopEdge  = 3;
  chanText[i].ITextFont = NULL;
  chanText[i].NextText = NULL;

   /* Now the gadget structures */

  if (i == 15)
    chanGadget[i].NextGadget = &pauseGadget;
  else
    chanGadget[i].NextGadget = &chanGadget[i+1];

  chanGadget[i].LeftEdge = gadgHposn + i * gadgHspacing;
  chanGadget[i].TopEdge = gadgVposn;
  chanGadget[i].Width = gadgetwidth;
  chanGadget[i].Height = gadgetheight;
  if (*(graph_chans+i))
    chanGadget[i].Flags =  GADGHBOX | SELECTED;
  else chanGadget[i].Flags = GADGHBOX;
  chanGadget[i].Activation = TOGGLESELECT|GADGIMMEDIATE;
  chanGadget[i].GadgetType = BOOLGADGET;
  chanGadget[i].GadgetRender = NULL;
  chanGadget[i].SelectRender = NULL;
  chanGadget[i].GadgetText = &chanText[i];
  chanGadget[i].MutualExclude = 0L;
  chanGadget[i].SpecialInfo = NULL;
  chanGadget[i].GadgetID = NULL;
  chanGadget[i].UserData = NULL;
  }
chanText[0].IText = (UBYTE*)"0";
chanText[1].IText = (UBYTE*)"1";
chanText[2].IText = (UBYTE*)"2";
chanText[3].IText = (UBYTE*)"3";
chanText[4].IText = (UBYTE*)"4";
chanText[5].IText = (UBYTE*)"5";
chanText[6].IText = (UBYTE*)"6";
chanText[7].IText = (UBYTE*)"7";
chanText[8].IText = (UBYTE*)"8";
chanText[9].IText = (UBYTE*)"9";
chanText[10].IText = (UBYTE*)"10";
chanText[11].IText = (UBYTE*)"11";
chanText[12].IText = (UBYTE*)"12";
chanText[13].IText = (UBYTE*)"13";
```

```
chanText[14].IText = (UBYTE*)"14";
chanText[15].IText = (UBYTE*)"15";
draw_display();
SetBPen(MyRPort,(int)black);
DataSwitch((short)TRUE);
ScreenFlag = TRUE;
}

void draw_display(void)
{

/* Draw the graph borders   */

SetRast(MyRPort,(int)graph_back_color);
midpt = highy + floor((double)(lowy-highy)/2.0);
SetAPen(MyRPort,(int)graph_color);
Move(MyRPort,(int)initx,(int)highy);
PolyDraw(MyRPort,5,&graphoutline[0]);
line(MyRPort,graph_color,initx,midpt,finalx,midpt);

/* Print the Y scale markers  */

SetAPen(MyRPort,(int)graph_color);
graphy = midpt - 10 * yscale + 3;
Move(MyRPort,(initx-30),(int)graphy);
Text(MyRPort,"+10",3);
graphy = midpt + 10 * yscale + 3;
Move(MyRPort,(initx-30),(int)graphy);
Text(MyRPort,"-10",3);
Move(MyRPort,initx-20,midpt+3);
Text(MyRPort,"0",1);

/* Draw ticks on y-axis at 1 volt spacing  */
numticks = 10;
for (i = 0;i <= numticks;i++)
  {
  graphy = midpt + i * yscale;
  line(MyRPort,graph_color,initx,graphy,(short)(initx-5),graphy);
  }
for (i = 1;i <= numticks;i++)
  {
  graphy = midpt - i * yscale;
  line(MyRPort,graph_color,initx,graphy,(short)(initx-5),graphy);
  }

/* Print the initial X scale markers   */

maxtime = samp_rate * (finalx - initx) * period;
xscale = (finalx - initx)/maxtime;
Move(MyRPort,(int)initx,lowy+15);
Text(MyRPort,"0",1);
PrintMaxTime();
numticks = 10;
for (i=1;i<numticks;i++)
  {
  graphx = initx + (maxtime/10.0) * xscale * i;
  line(MyRPort,graph_color,graphx,lowy,graphx,(short)(lowy+5));
  }
graphx = 0;

Move(MyRPort,(finalx - initx)/2 + initx - 20,lowy + 15);
SetAPen(MyRPort,(int)graph_color);
Text(MyRPort,"Time(sec)",9);
RefreshGadgets(&chanGadget[0],DispWind,NULL);

Move(MyRPort,M1CENTX-3,M1CENTY+3);
SetAPen(MyRPort,(int)yellow);
Text(MyRPort,"P",1);
Move(MyRPort,M2CENTX-3,M2CENTY+3);
SetAPen(MyRPort,(int)yellow);
Text(MyRPort,"F",1);
DrawDot(M1CENTX,M1CENTY-DOT_SPACE,0);
DrawDot(M1CENTX,M1CENTY+DOT_SPACE,0);
DrawDot(M1CENTX-DOT_SPACE,M1CENTY,0);
DrawDot(M1CENTX+DOT_SPACE,M1CENTY,0);
DrawDot(M2CENTX,M2CENTY-DOT_SPACE,0);
DrawDot(M2CENTX,M2CENTY+DOT_SPACE,0);
DrawDot(M2CENTX-DOT_SPACE,M2CENTY,0);
DrawDot(M2CENTX+DOT_SPACE,M2CENTY,0);
sprintf(TextBuf,"DataTime\n");
SetBPen(MyRPort,(int)black);
SetAPen(MyRPort,(int)white);
Move(MyRPort,DataTimeX,DataTimeY-10);
Text(MyRPort,TextBuf,8);
```

```
sprintf(TextBuf,"Runtime\n");
Move(MyRPort,RunTimeX,RunTimeY-10);
Text(MyRPort,TextBuf,7);
UpdateDisplays();
}

/*     Reset_graph() is called when the plot has reached the       */
/*     end of the window and needs to be scrolled left          */

void reset_graph(void)
{
short dx;

/*  Scroll the plot left  */

dx = scrollpt - initx - 1;
ScrollRaster(MyRPort,(int)dx,0,initx + 1,highy+1,finalx-1,lowy-1);

/*  Redraw the Graph outline  */

line(MyRPort,graph_color,graphx,midpt,finalx,midpt);
line(MyRPort,graph_color,graphx,highy,finalx,highy);
line(MyRPort,graph_color,graphx,lowy,finalx,lowy);
}

/* This routine checks the number of digits required (and hence
   the width) to print a number                              */

short checkdigs(double number)
{
if (number < 10.0)
  return(1);
if (number < 100.0)
  return(2);
if (number < 1000.0)
  return(3);
if (number < 10000.0)
  return(4);
return(0);
}
```

```
void calcerrors(void)
{
short i;
double a1,a2,a3,b1,b2,b3,c1,c2,c3,d1,d2,d3;

a1 = *(volts+4);
a2 = *(volts+5);
a3 = *(volts+6);
b1 = *(volts+7);
b2 = *(volts+8);
b3 = *(volts+9);
c1 = *(volts+10);
c2 = *(volts+11);
c3 = *(volts+12);
d1 = *(volts+13);
d2 = *(volts+14);
d3 = *(volts+15);
/**********************************************************************
   If the alternate channels are selected (gadget AND switch on the back
   of the pyroelectric control circuitry) then the laser power,current
   and voltage are read at channels 0 thru 2. These values must be scaled
   appropriately to correspond to true values.
**********************************************************************/
if (pivflag)
 {
 *(volts+0) = *(volts+0) * PwrFactor + PwrOffset;  /* output power */
 *(volts+1) = *(volts+1) * CurFactor + CurOffset;  /* discharge current */
 *(volts+2) = *(volts+2) * VolFactor + VolOffset;  /* discharge voltage */
 }
/**********************************************************************
    calculate a sum of all the sensor signals
    to obtain a measure proportional to total power
**********************************************************************/
sumtotal = -(a1+a2+a3+b1+b2+b3+c1+c2+c3+d1+d2+d3);
sumtotal = pyrooffset + sumtotal * pyrofactor;
if (sumtotal < 0)
 sumtotal = 0.2;

/**********************************************************************
   calculate uniformity and position error
   signals
**********************************************************************/
YPosErr = (a3+b3+c1+d1)-(a1+b1+c3+d3)+yoffsetp;
XPosErr = (a3+d3+b1+c1)-(a1+d1+b3+c3)+xoffsetp;
YPosErr *= Kcyp;
XPosErr *= Kcxp;

YUnifErr = (a1+a2+a3+b1+b2+b3)-(c1+c2+c3+d1+d2+d3);
XUnifErr = (a1+a2+a3+d1+d2+d3)-(b1+b2+b3+c1+c2+c3);
YUnifErr *= Kcyu;
XUnifErr *= Kcxu;

/*****************************************************************
  Assign power variable used for output power stabilization
******************************************************************/

if (pivflag)
 {
 power = *(volts+0);
 current = *(volts+1);
 voltage = *(volts+2);
 *(volts+1) = current - 15.0;
 }
else
 power = PwrSetpt;
}

/*****************************************************************
  cntup increments the counters used in
  the digital filtering routines
*****************************************************************/

void cntup(void)
{
n++;   /* n */
nm1++;  /* n-1 */
nm2++;  /* n-2 */
nm3++;  /* n-3 */

if(n > 3)   n = 0;
if(nm1>3) nm1 = 0;
if(nm2>3) nm2 = 0;
if(nm3>3) nm3 = 0;
}

/*****************************************************************
  filter() implements a digital filter (obtained
  by analog to digital transformation of a 3rd
  order butterworth filter) Various different cutoff
  frequencies are used depending on the requirements
*****************************************************************/

void filter(float *input,float *output,float *ybuf,float *xbuf,int fn)
{
*(xbuf+n) = *input;
*(ybuf+n) = a1[fn] * *(xbuf+n) + a2[fn] * *(xbuf+nm1) + a3[fn] * *(xbuf+nm2) +
        a4[fn] * *(xbuf+nm3) - b2[fn] * *(ybuf+nm1) - b3[fn] * *(ybuf+nm2) -
        b4[fn] * *(ybuf+nm3);
*output = *(ybuf+n);
}

/*****************************************************************
  filter_all() calls the filter() routine to perform
  filtering of the various signals in the system
*****************************************************************/

void filter_all(void)
{
if (filtflag)
 {
 for (i=0;i<16;i++)
  filter(volts+i,filtvolts+i,(float*)&ybuf[i][0],(float*)&xbuf[i][0],CHANFILTER);

filter((float*)&sumtotal,(float*)&sumtotal,(float*)&ybuf[16][0],(float*)&xbuf[16][0],CHA
NFILTER);
 }
filter((float*)&XPosErr,(float*)&XPosErr,(float*)&ybuf[17][0],(float*)&xbuf[17][0],POS
NFILTER);
filter((float*)&YPosErr,(float*)&YPosErr,(float*)&ybuf[18][0],(float*)&xbuf[18][0],POS
NFILTER);
filter((float*)&XUnifErr,(float*)&XUnifErr,(float*)&ybuf[19][0],(float*)&xbuf[19][0],UNI
FFILTER);
filter((float*)&YUnifErr,(float*)&YUnifErr,(float*)&ybuf[20][0],(float*)&xbuf[20][0],UNI
FFILTER);
filter((float*)&power,(float*)&power,(float*)&ybuf[21][0],(float*)&xbuf[21][0],POWER
FILTER);
cntup();
}

/*****************************************************************
```

```
   This routine is used to confine plotted signals within
   the vertical boundries of the plotting area
*****************************************************************/
void clip(float *ip)
{
if (*ip >= 9.8) *ip = 9.8;
if (*ip <= -9.8) *ip = -9.8;
}

/****************************************************************
   This routine calculates digital filter coefficients for a 3rd
   order Butterworth low-pass filter with a given cutoff freq. 'fc'
   or for a simple, first order, low-pass filter
****************************************************************/
void assign_coefs(float fcutoff,int filtnum,short FilterType)
{
double wc,wc2,wc3;
double ts,ts2,ts3;

if (fcutoff < 0.01)
  {
  quit = TRUE;
  printf("ATTEMPT TO ASSIGN INCORRECT FILTER COEFFICIENT\n");
  printf("MINIMUM ALLOWED CUTOFF FREQUENCY = 0.01 Hz.\n");
  printf("FILTER NUMBER = %d\n",filtnum);
  }
else if (FilterType == BUTTERWORTH)
  {
  wc = 2 * PI * fcutoff;
  wc2 = wc * wc;
  wc3 = wc2 * wc;
  ts = period;
  ts2 = ts * ts;
  ts3 = ts2 * ts;

  a1[filtnum] = wc3;
  a2[filtnum] = 3 * wc3;
  a3[filtnum] = 3 * wc3;
  a4[filtnum] = wc3;

  b1[filtnum] = 8/ts3 + 8*wc/ts2 + 4*wc2/ts + wc3;
  b2[filtnum] = -24/ts3 - 8*wc/ts2 + 4*wc2/ts + 3*wc3;
  b3[filtnum] = 24/ts3 - 8*wc/ts2 - 4*wc2/ts + 3*wc3;
  b4[filtnum] = -8/ts3 + 8*wc/ts2 - 4*wc2/ts + wc3;

  a1[filtnum] = a1[filtnum]/b1[filtnum];
  a2[filtnum] = a2[filtnum]/b1[filtnum];
  a3[filtnum] = a3[filtnum]/b1[filtnum];
  a4[filtnum] = a4[filtnum]/b1[filtnum];

  b2[filtnum] = b2[filtnum]/b1[filtnum];
  b3[filtnum] = b3[filtnum]/b1[filtnum];
  b4[filtnum] = b4[filtnum]/b1[filtnum];
  b1[filtnum] = 1.0;
  }
else if (FilterType == FIRSTORDER)
  {
  wc = 2 * PI * fcutoff;
  ts = period;

  b1[filtnum] = 1.0 + 2.0/(wc*ts);
  b2[filtnum] = 1.0 - 2.0/(wc*ts);
  b3[filtnum] = 0;
  b4[filtnum] = 0;

  a1[filtnum] = 1/b1[filtnum];
  a2[filtnum] = 1/b1[filtnum];
  a3[filtnum] = 0;
  a4[filtnum] = 0;

  b2[filtnum] = b2[filtnum]/b1[filtnum];
  b1[filtnum] = 1.0;
  }
}

/*****************************************************************
   This routine can be used to toggle the selection state
   of a gadget from within a program
*****************************************************************/
void toggleGadget(struct Gadget* gadget)
{
int posn;

posn = RemoveGadget(DispWind,gadget);
```

```
RefreshGadgets(gadget,DispWind,NULL);
gadget->Flags ^= SELECTED;
AddGadget(DispWind,gadget,posn);
RefreshGadgets(gadget,DispWind,NULL);
}

/**************************************************************
    read_speeds() reads the micrometer speeds (previously stored
    in micspeeds.pars by the program "setspeed"
**************************************************************/
void read_speeds(void)
{
char parval1[10];
char parval2[10];
char comment[100];
int input;

if ((paramfp = fopen("micspeeds.pars","r")) == 0)
  {
  printf("Can't open speed settings file! (micspeeds.pars)\n");
  quit = TRUE;
  }
if (!quit)
  {
  for (i=1;i<=4;i++)
    {
    fgets(comment,100,paramfp);
    fgets(comment,100,paramfp);
    fgets(comment,8,paramfp);
    fgets(parval1,5,paramfp);
    fgets(comment,50,paramfp);
    fgets(comment,8,paramfp);
    fgets(parval2,5,paramfp);
    fgets(comment,50,paramfp);
    MicFwdSpeed[i] = atof(parval1);
    MicRevSpeed[i] = atof(parval2);
    MicSpeed[i] = (MicFwdSpeed[i]+MicRevSpeed[i])/2.0;
    }
  }
  /*
 for (i=1;i<=4;i++)
  printf("mic%d fwd = %f  rev = %f\n",i,MicFwdSpeed[i],MicRevSpeed[i]);
```

```
  */
fclose(paramfp);
}


void SetupDatafile(void)
{
/*****************************************************************
   open a file to store data in (on the ram disk)
*****************************************************************/
if ((datafp = fopen("ram:datafile","w")) == 0)
   {
   printf("Can't open data file, aborting!\n");
   exit(1);
   }
/*
fprintf(datafp,"Sampling period = %f\r\n",samp_rate*period);
*/
DataTimeOffset = runtime;
}

/***************************************************************
   -PwrControl() implements a standard discrete proportional
    controller which stabilizes the laser output power using
    feedback from the pyroelectric sensor array
   -The output from this controller is fed back to the laser
    sustainer current setpoint via the Proto-40K digital to
    analog converter output
***************************************************************/
void PwrControl(void)
{
if (pivflag & PwrAuto)
  {
  Ept = PwrSetpt - power;
  Upwr += KcPwr * (Ept-Eptm1 + TsPwr*Ept/TiPwr);
  Eptm1 = Ept;
  if (Upwr < 0)
    Upwr = 0;
  if (Upwr > UPMAX)
    Upwr = UPMAX;
  }
 else
 Upwr = IdleSetpt;

UpwrPlot = Upwr - 7.5;

DtoA = Upwr * 255/15;
*p40_da1 = DtoA;
}


/*******************************************************************
    DataSwitch() is used to select either the Power meter,Discharge
    voltage and current OR the uniformity and position error signals
    (vertical and horizontal).  The switch on the rear of the
    alignment controller circuits should be set to match the setting
    of the PIV gadget
*******************************************************************/
void DataSwitch(short PIVOnOff)
{
if (!PIVOnOff)
 {
 if (PwrAuto)
  {
  PwrAuto = FALSE;
  toggleGadget(&PAUTOGadget);
  }
 }
chanText[0].FrontPen = black;
chanText[1].FrontPen = black;
chanText[2].FrontPen = black;
RefreshGadgets(&chanGadget[0],DispWind,NULL);
if (!PIVOnOff)
 {
 chanText[0].IText = (UBYTE*)"0";
 chanText[1].IText = (UBYTE*)"1";
 chanText[2].IText = (UBYTE*)"2";
 }
else if (PIVOnOff)
 {
 chanText[0].IText = (UBYTE*)"P";
 chanText[1].IText = (UBYTE*)"I";
 chanText[2].IText = (UBYTE*)"V";
 }
chanText[0].FrontPen = chan_cols[0];
chanText[1].FrontPen = chan_cols[1];
chanText[2].FrontPen = chan_cols[2];
RefreshGadgets(&chanGadget[0],DispWind,NULL);
}


void PrintMaxTime(void)
{
SetAPen(MyRPort,(int)graph_color);
maxtime = samp_rate * (finalx - initx) * period;
NewMaxTime = maxtime;
Move(MyRPort,finalx-25,lowy+15);
sprintf(TextBuf,"%5d\n",NewMaxTime);
Text(MyRPort,TextBuf,5);
}

void UpdateDisplays(void)
{
/***************************************************************
   The data timer displays the elapsed time since data storage
   was initiated
***************************************************************/
if (file_data)
 {
 Secs = 60 * modf((runtime-DataTimeOffset)/60,&Mins);
 shortMins = Mins;
 }
else
 {
 Secs = 0;
 shortMins = 0;
 }
if (Secs < 10)
 sprintf(TextBuf,"%3d:0%1d\n",shortMins,Secs);
else
 sprintf(TextBuf,"%3d:%2d\n",shortMins,Secs);
Move(MyRPort,DataTimeX,DataTimeY);
SetBPen(MyRPort,(int)orange);
SetAPen(MyRPort,(int)black);
Text(MyRPort,TextBuf,6);
/***************************************************************
```

```
  The data timer displays the program began running
  NOTE: The timer is stopped when pause is selected
*************************************************************/
Secs = 60 * modf(runtime/60,&Mins);
shortMins = Mins;
if (Secs < 10)
  sprintf(TextBuf,"%3d:0%1d\n",shortMins,Secs);
else
  sprintf(TextBuf,"%3d:%2d\n",shortMins,Secs);
Move(MyRPort,RunTimeX,RunTimeY);
Text(MyRPort,TextBuf,6);

if (!pivflag)
  SetBPen(MyRPort,(int)graph_back_color);
sprintf(TextBuf,"P=%4.1f kW\n",power);
Move(MyRPort,PwrX,PwrY);
Text(MyRPort,TextBuf,9);
sprintf(TextBuf,"I=%4.1f A \n",current);
Move(MyRPort,CurX,CurY);
Text(MyRPort,TextBuf,9);
sprintf(TextBuf,"V=%4.1f kV\n",voltage);
Move(MyRPort,VolX,VolY);
Text(MyRPort,TextBuf,9);

SetBPen(MyRPort,(int)graph_back_color);
}
```

CONTROL.H

```
/*************************************************************************

CONTROL.H CONTAINS VARIABLES AND DEFINITIONS USED ONLY IN
CONTROL.C (PI4:CONTROL/CONTROL.C)
*************************************************************************/

/* variables used in position control routine */

double TposMin,TposMax,Tposx,Tposy;  /* sampling intervals for position control */
double YCntrlTime,XCntrlTime,cntrltime;  /* time of last control interval */
double paramtime;
double PosnPulseLen;
double MicPulseLen[5];
float Kcyp,Kcxp;
float Kcyu,Kcxu;
float fcposn,fcchan,fcunif; /* different cutoff frequencies used */
float xposntime,yposntime;
float YPosErr,XPosErr;
short OldYPosErr,OldXPosErr;
float YUnifErr,XUnifErr;
short OldYUnifErr,OldXUnifErr;
float ypo_pulse,xpo_pulse;
float yun_pulse,xun_pulse;
double YunifTime,XunifTime;
float yautogainp,xautogainp;
float yautogainu,xautogainu;
float autofactorp,autofactoru;
float yoffsetu,xoffsetu;
float yoffsetp,xoffsetp;
short gainpflag, gainuflag;
short movingyposn,movingxposn;
short AxisFlag;
short CheckSpeedFlag;
double tempxerr;
double tempyerr;
#define VERT 0
#define HORZ 1
```

```
#define M1CENTX (int)350
#define M1CENTY (int)40
#define M2CENTX (int)300
#define M2CENTY (int)40
#define DOT_SPACE (int)15

/* variables used in uniformity identification program */

short MicDirn;
double IdTime0,PhaseTime[6],PhaseDist[6];
short CurMic,CurPhase,PhaseDirn[6],CurDist;
short IdFlag;
short MicStopped;
float MicFwdSpeed[5],MicRevSpeed[5];
float MicSpeed[5];

/* variables used in power control */

UBYTE DtoA; /* used for D/A output */
double power;
double PwrTime;
double TsPwr;
double KcPwr,TIPwr;
double Ept,Eptm1,Upwr,UpwrPlot;
short OldUpwrPlot;
double fcpwr;
double PwrSetpt;
short OldPwrSetpt;
double IdleSetpt;
short PwrAuto;
double UPMAX;

/* scale factors used to scale data */
float PwrFactor;
float CurFactor;
float VolFactor;
float PwrOffset;
float CurOffset;
float VolOffset;
float pyrofactor;
float pyrooffset;
float chan_gain[16];
```

```
float temp_gain[4];
short chanfilt[16]; /* stores which filter number will be used to filter
                        each channel */
#define CHANFILTER 1
#define POSNFILTER 2
#define POWERFILTER 3
#define UNIFFILTER 4
#define BUTTERWORTH (short)0
#define FIRSTORDER (short)1
short CurFiltType;

/* used to store digital filter coefficients */

double a1[6],a2[6],a3[6],a4[6];
double b1[6],b2[6],b3[6],b4[6];

/* used to store analog data from proto-40k */
float *volts;
float *volts_avgs;
float *filtvolts;
float sumtotal;
float current,voltage;

/* definitions */

#define PI 3.141592654
#define mic1 (short)1
#define mic2 (short)2
#define mic3 (short)3
#define mic4 (short)4
#define on (short)0
#define off (short)1
#define GO (short)1
#define STOP (short)0
#define fwd (short)1
#define rev (short)0
/***********************************************************************
  The following definitions may be used to obtain
  correct direction of micrometer movements if the
  control actions of the alignment control system
  are channelled through the computer
***********************************************************************/
```

```
#define m1_up fwd
#define m1_down rev
#define m2_up fwd
#define m2_down rev
#define m1_rt rev
#define m1_left fwd
#define m2_rt fwd
#define m2_left rev
/*********************************************************************
   define values to operate piggyback board designed
   by Chris Sellathamby for computer/motormic interface
*********************************************************************/
#define ld_cnt_mode 0x4000
#define enable 0x8000
#define disable 0x7fff
#define cntrl_latch 0x2000
#define no_latch 0x9fff
#define enable_cntr 0xffef
#define ld_cntr 0x0100
#define r1_fwd 0x0
#define r1_rev 0x81
#define r2_fwd 0x2
#define r2_rev 0x83
#define r3_fwd 0x4
#define r3_rev 0x85
#define r4_fwd 0x6
#define r4_rev 0x87
#define clr_muxaddr 0xfff8
#define start_ld_clk 0x0200
#define end_ld_clk 0xfdff
#define start_dclk 0x20
#define end_dclk 0xffdf
#define start_cclk 0x200
#define end_cclk 0xfdff
#define set_ff 0x40
#define reset_ff 0xffbf
#define clr_op_latch 0x0400
#define addr_latch_mode 0x800
#define demux_mode 0x0
#define mem_mode 0x0c00
#define use_normal_io 0x6000
#define start_pdwn 0x0008
#define stop_pdwn 0xfff7
#define start_pup 0x0010
#define stop_pup 0xffef
#define change_mode 0xf3ff


#define pi 3.141592654
#define bit0 (short)1
#define bit1 (short)2
#define bit2 (short)4
#define bit3 (short)8
#define bit4 (short)16
#define bit5 (short)32
#define bit6 (short)64
#define bit7 (short)128
#define bit8 (short)256
#define bit9 (short)512
#define bit10 (short) 1024
#define bit11 (short) 2048
#define bit12 (short) 4096
#define bit13 (short) 8192
#define bit14 (short) 16384
#define bit15 (short) 32768
#define last_chan (short) 16
#define maxcnt  (long)200000 /*checks if wand is running (read_data())*/
#define avgnmbr (short)10    /*number of samples taken from each analog channel
to minimize noise corruption*/
#define control_on (short)1
#define control_off (short)0

/********************************************************
   Global variable declarations
********************************************************/
short i,j;
short quit;
short no_syncflag;
int end;
short m1down,m1up,m1rt,m1left;
short m2down,m2up,m2rt,m2left;
short m1vert,m1horz,m2vert,m2horz;
short pauseflag;
short run_mins, run_secs;
short diginput,temp_dinput, ok_flag;
short abort_control;
short file_data,old_file_data;
short first_time;
short num_samps;
short ad_data;
short test_dist;
short cur_mic;
double MicBacklash[5];
short MicMoving[5];
short BackCompDone;
short OldMicDirn[5];
short mic_cntr_off[5];
short mic_cntr[5];
short OldMicCount;
short samp_rate,OldSampRate;
short MicIp;
short selected_chans[16];
short samp_cntr;
short M1auto;
short M2auto;

unsigned short control_word;
unsigned short temp_cntrl_word;

char storedata_flag;
char *strbuf;

/* variables used in time keeping */

struct timeval cur_time;
double last_time,time1,time0;
double high_limit,low_limit;
double stop_time,start_time,time_offset;
double samptime;
double period;
double runtime,updatetime,oldtime,DataTimeOffset;
short Secs,shortMins;
double Mins;

/* File handling declarations */

FILE *fopen();
FILE *datafp;
FILE *paramfp;
```

CONTROL.PAR

```
PwrSetpt            3.5       Setpoint for Power control
(kW)
IdleSetpt           7.5       Idle Setpoint for Power control
(volts)
KcPwr               1         Kc for Power Controller
TiPwr               2.5       Ti for Power Controller
TsPwr               0.5       Ts for Power Controller
samp_rate           3         Integer Data storage interval (*
0.1 secs)
TposMax             7.0       Max sampling period for
position control
TposMin             2         Min sampling period for
position control
PosnPulseLen        0.25      Pulse Length for Position
Control
Kcyp                1.2       position controller gain (vert)
Kcxp                1.5       position controller gain (horz)
Kcyu                0.35      uniformity controller gain (vert)
Kcxu                0.38      uniformity controller gain (horz)
yoffsetp            0         position error signal offset
(vert)
xoffsetp            0.15      position error signal offset
(horz)
yoffsetu            0.5       uniformity error offset (vert)
xoffsetu            1         uniformity error offset (horz)
fcposn              0.03      cutoff freq. for position error
signals
fcunif              0.1       cutoff freq. for uniformity error
signals
fcchan              0.1       cutoff freq. for analog data
channels
fcpwr               0.8       cutoff freq. for power control
CurFiltType         1         (0-3rd order Butterworth, 1-
simple first order)
PwrFactor           -3.2      scaling factor for laser power
meter (kWatt/Volt)
CurFactor           7.23      scaling factor for discharge
current (Amps/Volt)
VolFactor           0.86      scaling factor for discharge
voltage (kV/Volt)
PwrOffset           0.1       Offset for laser power meter
(dump)
CurOffset           0         Offset for discharge current
VolOffset           0         Offset for discharge voltage
pyrofactor          -0.185    converts sensor signals to
equivalent power in kw.
pyrooffset          -.8       corrects for sensor offset
backlash            5 5 5 1   Micrometer backlash for the 4
micrometers

NOTE: THIS FILE IS IN THE ORDER EXPECTED BY THE READ_PARS()
ROUTINE, DO NOT CHANGE IT OR INCORRECT PARAMETER VALUE
ASSIGNMENTS WILL RESULT
```

SUBS.C

```
/*****************************************************************
Subs.c contains all the subroutines called by pi4:control/control.c

Most of these routines operate in conjunction with the proto-40k data-acq. board
and the piggyback board designed by Chris Sellathamby during his PhD project.

I don't claim to be a software engineer so new users may find my code difficult to
understand.  The routines have all been used extensively and I have found no
bugs thus far.

                                                    Dale Akitt
*****************************************************************/
#include <stdio.h>
#include <math.h>
#include <functions.h>
#include <exec/memory.h>
#include <devices/timer.h>
#include <intuition/intuition.h>
#include "p40.h"
#include "control.h"

void pause(double period);
void move_mic(short mic,short stop_go,short dirn);
void control(void);
void read_data(void);
void reset_timer(void);
void update_timer(void);
void start_timer(void);
void measure_period(void);
void all_stop(void);
void update_cntr(short mic,short dirn,short dist);
void run_mic(short mic,short dirn,short dist);
void BackComp(short mic,short newdirn);
void DrawDot(int x,int y,short on_off);
void line(struct RastPort *RPort,short color,short x0,short y0,
    short x1,short y1);

/*************************************************
   pause provides a pause of length "period" where
   period is in seconds
*************************************************/
void pause(double period)
{
double time0,time1;
get_time1(&cur_time);
time0 = cur_time.tv_secs + cur_time.tv_micro/1e6;
while(time1-time0 < period)
   {
   get_time1(&cur_time);
   time1 = cur_time.tv_secs + cur_time.tv_micro/1e6;
   }
}

/**************************************************
   move_mic() is used to stop or start a micrometer
   running in a specified direction (via the piggyback
   board designed by Chris Sellathamby)
**************************************************/
void move_mic(short mic,short stop_go,short dirn)
{
/*
if (dirn)
  printf("moving mic fwd\n");
else if (!dirn)
  printf("moving mic rev\n");
*/

switch(mic)
  {
  case mic1:
   if (dirn == m2_up)
     DrawDot(M2CENTX,M2CENTY-DOT_SPACE,stop_go);
   else if (dirn == m2_down)
     DrawDot(M2CENTX,M2CENTY+DOT_SPACE,stop_go);
   break;
  case mic2:
   if (dirn == m2_rt)
     DrawDot(M2CENTX+DOT_SPACE,M2CENTY,stop_go);
   else if (dirn == m2_left)
```

```
      DrawDot(M2CENTX-DOT_SPACE,M2CENTY,stop_go);
    break;
  case mic3:
    if (dirn == m1_up)
      DrawDot(M1CENTX,M1CENTY-DOT_SPACE,stop_go);
    else if (dirn == m1_down)
      DrawDot(M1CENTX,M1CENTY+DOT_SPACE,stop_go);
    break;
  case mic4:
    if (dirn == m1_rt)
      DrawDot(M1CENTX+DOT_SPACE,M1CENTY,stop_go);
    else if (dirn == m1_left)
      DrawDot(M1CENTX-DOT_SPACE,M1CENTY,stop_go);
    break;
  }

control_word &= disable;
control_word &= no_latch;
control_word |= (cntrl_latch | mem_mode);
control_word &= clr_muxaddr;
*p40_doutput = control_word;
control_word |= enable;
*p40_doutput = control_word;
control_word &= disable;
*p40_doutput = control_word;
switch (mic)
   {
    case 1:
      if (dirn) control_word |= r1_fwd;
      else control_word |=  r1_rev;
      *p40_doutput = control_word;
      break;
    case 2:
      if (dirn) control_word |=  r2_fwd;
      else    control_word |=  r2_rev;
      *p40_doutput = control_word;
      break;
    case 3:
      if (dirn) control_word |= r3_fwd;
      else control_word |=  r3_rev;
      *p40_doutput = control_word;
      break;
    case 4:
      if (dirn) control_word |= r4_fwd;
      else control_word |=  r4_rev;
      *p40_doutput = control_word;
      break;
    }
switch (stop_go)        /* case 0 -> stop */
    {
    case 0:
      MicMoving[mic] = FALSE;
      control_word &= reset_ff;
      *p40_doutput = control_word;
      control_word |= enable;
      *p40_doutput = control_word;
      control_word |= start_dclk;
      *p40_doutput = control_word;
      control_word &= end_dclk;
      *p40_doutput = control_word;
      break;
    case 1:             /* case 1 -> go */
      MicMoving[mic] = TRUE;
      control_word |= set_ff;
      *p40_doutput = control_word;
      control_word |= enable;
      *p40_doutput = control_word;
      control_word |= start_dclk;
      *p40_doutput = control_word;
      control_word &= end_dclk;
      *p40_doutput = control_word;
      break;
    }
control_word &= disable;
*p40_doutput = control_word;
control_word &= change_mode;
control_word |= addr_latch_mode;
*p40_doutput = control_word;
control_word |= enable;
*p40_doutput = control_word;
control_word &= disable;
*p40_doutput = control_word;
control_word |= mem_mode;
*p40_doutput = control_word;
```

```
control_word &= disable;
*p40_doutput = control_word;
}

/****************************************************************************
    This routine controls beam position by simultaneously
    moving both the primary and feedback mirrors (pie4)
****************************************************************************/

extern struct Gadget DataGadget;
extern struct Gadget M1AUTOGadget;

void control(void)
{
/********************         NOTE      ********************
   To change directions of movements, change values of m1_up
   etc in control.h
*******************************************************************/
if (!IdFlag)
  {
  m1up = (diginput & bit12) >> 12;
  m1down = (diginput & bit13) >> 13;
  m1left = (diginput & bit14) >> 14;
  m1rt = abs((diginput & bit15) >> 15);

  if (M2auto)
   {
   if (movingyposn)
    {
    if ((runtime-yposntime) > ypo_pulse)
     {
     move_mic(mic1,STOP,fwd);
     move_mic(mic1,STOP,rev);
     move_mic(mic3,STOP,fwd);
     move_mic(mic3,STOP,rev);
     movingyposn = FALSE;
     }
    }
   else if ((runtime-YCntrlTime) >= Tposy)
    {
    YCntrlTime = runtime;
    Tposy = TposMax + fabs(YPosErr)/10.0 *(TposMin-TposMax) ;
```

```
    if (YPosErr > 0.5)
     {
     movingyposn = TRUE;
     yposntime = runtime;
     ypo_pulse = PosnPulseLen;
     move_mic(mic1,GO,fwd);
     move_mic(mic3,GO,rev);
     }
    else if (YPosErr < -0.5)
     {
     movingyposn = TRUE;
     yposntime = runtime;
     ypo_pulse = PosnPulseLen;
     move_mic(mic1,GO,rev);
     move_mic(mic3,GO,fwd);
     }
    }

   if (movingxposn)
    {
    if ((runtime-xposntime) > xpo_pulse)
     {
     move_mic(mic2,STOP,fwd);
     move_mic(mic2,STOP,rev);
     move_mic(mic4,STOP,fwd);
     move_mic(mic4,STOP,rev);
     movingxposn = FALSE;
     }
    }
   else if ((runtime-XCntrlTime) >= Tposx)
    {
    Tposx = TposMax + fabs(XPosErr)/10.0 * (TposMin-TposMax);
    XCntrlTime = runtime;
    if (XPosErr > 0.5)
     {
     movingxposn = TRUE;
     xposntime = runtime;
     xpo_pulse = PosnPulseLen;
     move_mic(mic2,GO,fwd);
     move_mic(mic4,GO,fwd);
     }
    else if (XPosErr < -0.5)
   {
   movingxposn = TRUE;
   xposntime = runtime;
   xpo_pulse = PosnPulseLen;
   move_mic(mic2,GO,rev);
   move_mic(mic4,GO,rev);
   }
  }
 }
if (M1auto)
 {
 if (!movingyposn)
  {
  if (!*(MicMoving+mic3))
   {
   if (m1up)
    {
    if (*(OldMicDirn+mic3) != m1_up)
     yun_pulse = 0.08 + *(MicBacklash+mic3)/ *(MicSpeed+mic3);
    else
     yun_pulse = 0.08;
    YunifTime = runtime;
    move_mic(mic3,GO,m1_up);
    *(MicMoving+mic3) = TRUE;
    *(OldMicDirn+mic3) = m1_up;
    }
   else if (m1down)
    {
    if (*(OldMicDirn+mic3) != m1_down)
     yun_pulse = 0.08 + *(MicBacklash+mic3)/ *(MicSpeed+mic3);
    else
     yun_pulse = 0.08;
    YunifTime = runtime;
    move_mic(mic3,GO,m1_down);
    *(MicMoving+mic3) = TRUE;
    *(OldMicDirn+mic3) = m1_down;
    }
   }
  else if ((runtime-YunifTime) > yun_pulse)
   {
   move_mic(mic3,STOP,*(OldMicDirn+mic3));
   *(MicMoving+mic3) = FALSE;
```

```
   if (m1up)
    {
    if (*(OldMicDirn+mic3) != m1_up)
     yun_pulse = 0.08 + *(MicBacklash+mic3)/ *(MicSpeed+mic3);
    else
     yun_pulse = 0.08;
    YunifTime = runtime;
    move_mic(mic3,GO,m1_up);
    *(MicMoving+mic3) = TRUE;
    *(OldMicDirn+mic3) = m1_up;
    }
   else if (m1down)
    {
    if (*(OldMicDirn+mic3) != m1_down)
     yun_pulse = 0.08 + *(MicBacklash+mic3)/ *(MicSpeed+mic3);
    else
     yun_pulse = 0.08;
    YunifTime = runtime;
    move_mic(mic3,GO,m1_down);
    *(MicMoving+mic3) = TRUE;
    *(OldMicDirn+mic3) = m1_down;
    }
   }
  }

 if (!movingxposn)
  {
  if (!*(MicMoving+mic4))
   {
   if (m1rt)
    {
    if (*(OldMicDirn+mic4) != m1_rt)
     yun_pulse = 0.08 + *(MicBacklash+mic4)/ *(MicSpeed+mic4);
    else
     yun_pulse = 0.08;
    YunifTime = runtime;
    move_mic(mic4,GO,m1_rt);
    *(MicMoving+mic4) = TRUE;
    *(OldMicDirn+mic4) = m1_rt;
    }
   else if (m1left)
    {
      if (*(OldMicDirn+mic4) != m1_left)
        yun_pulse = 0.08 + *(MicBacklash+mic4)/ *(MicSpeed+mic4);
      else
        yun_pulse = 0.08;
      YunifTime = runtime;
      move_mic(mic4,GO,m1_left);
      *(MicMoving+mic4) = TRUE;
      *(OldMicDirn+mic4) = m1_left;
      }
    }
  else if ((runtime-YunifTime) > yun_pulse)
    {
    move_mic(mic4,STOP,*(OldMicDirn+mic4));
    *(MicMoving+mic4) = FALSE;
    if (m1rt)
      {
      if (*(OldMicDirn+mic4) != m1_rt)
        yun_pulse = 0.08 + *(MicBacklash+mic4)/ *(MicSpeed+mic4);
      else
        yun_pulse = 0.08;
      YunifTime = runtime;
      move_mic(mic4,GO,m1_rt);
      *(MicMoving+mic4) = TRUE;
      *(OldMicDirn+mic4) = m1_rt;
      }
    else if (m1left)
      {
      if (*(OldMicDirn+mic4) != m1_left)
        yun_pulse = 0.08 + *(MicBacklash+mic4)/ *(MicSpeed+mic4);
      else
        yun_pulse = 0.08;
      YunifTime = runtime;
      move_mic(mic4,GO,m1_left);
      *(MicMoving+mic4) = TRUE;
      *(OldMicDirn+mic4) = m1_left;
      }
    }
   }
  }
 }
}
```

```
/*****************************************************
   -read_data reads the analog channels which are to be
    sampled and places the values in the array "volts"
   -the digital inputs are also read and used later to
    transfer control from the auto-align circuit to
    the alignment micrometer drivers
   -the entire operation is synchronized to the
    rotating wand by reading bit8 of the digital
    inputs
*****************************************************/
void read_data(void)
{
short i,j;
long cnt;

control_word &= disable;
*p40_doutput = control_word;
temp_cntrl_word = use_normal_io;
*p40_doutput = temp_cntrl_word;
temp_cntrl_word != enable;
*p40_doutput = temp_cntrl_word;

diginput = *p40_dinput;
cnt = 0;
while ((diginput & bit8) == 0)
   {
   diginput = *p40_dinput;
   cnt++;
   if (cnt > maxcnt)
     {
     diginput = bit8;
     all_stop();
     no_syncflag = 1;
     }
   }
 cnt = 0;
 if (!no_syncflag)
   while ((diginput & bit8) == bit8)
    {
    diginput = *p40_dinput;
    cnt++;
    if (cnt > maxcnt)
```

```
      {
      diginput = 0;
      all_stop();
      no_syncflag = 1;
      }
    }
/****************************************************
   Read the digital channels 5 times in a row to
   ensure they have not been noise corrupted
****************************************************/
ok_flag = FALSE;
while(ok_flag == FALSE)
   {
   ok_flag = TRUE;
   diginput = *p40_dinput;
   for (i=0;i<5;i++)
      {
      temp_dinput = *p40_dinput;
      if (temp_dinput == diginput);
      else ok_flag = FALSE;
      }
   }
/***   Read the analog channels   ***/
for (i=0;i<last_chan;i++)
   *(volts+i) = 0;
for (j=0;j<avgnmbr;j++)
   {
   for (i=0;i<last_chan;i++)
      if (*(selected_chans+i)==1)
         {
         *p40_mux = i<<2;
         *p40_strobe = 0;
         while (*p40_end >=0);
         ad_data= *p40_data & 0x0fff;
         *(volts+i) += (ad_data - 2048)/204.8;
         }
   }
for (i=0;i<last_chan;i++)
   {
   if (*(selected_chans+i) == 1)
   *(volts+i) /= avgnmbr;
   }
```

```
temp_cntrl_word &= disable;
*p40_doutput = temp_cntrl_word;
}

/*******************************************************
   The following four routines are responsible for
   timekeeping during the identification run
   The names are self explanatory
*******************************************************/
void reset_timer(void)
{
get_time1(&cur_time);
time0 = cur_time.tv_secs + cur_time.tv_micro/1e6;
stop_time = time0;
time_offset = 0;
runtime = 0;
}

void stop_timer(void)
{
stop_time = last_time;
}

void start_timer(void)
{
read_data();
get_time1(&cur_time);
start_time = cur_time.tv_secs + cur_time.tv_micro/1e6;
time_offset += start_time - stop_time;
}

void update_timer(void)
{
get_time1(&cur_time);
time1 = cur_time.tv_secs + cur_time.tv_micro/1e6;
runtime = time1 - time0 - time_offset;
last_time = time1;
}

/*** Measure_period() is called at the
   beginning of the run to obtain an initial
   estimate of the rotating wands period      ***/
```

```
void measure_period(void)
{
short i;
short numrevs = 20;
double dt;
double t0;

update_timer();
stop_timer();
t0 = runtime;

start_timer();
for (i=0;i<numrevs;i++)
   {
   if (!no_syncflag)
   read_data();
   update_timer();
   }
stop_timer();
dt = runtime - t0;
period = dt/numrevs;
if (!no_syncflag)
  {
  printf("  Ts = %5.2f ms  Wand Speed = %4.0f RPM\r",1000 *
period,1/period*60.0);
  fflush(stdout);
  }
start_timer();
}

/*********************************************************
   all_stop() stops all micrometers
*********************************************************/

void all_stop(void)
{
move_mic(mic1,STOP,fwd);
move_mic(mic1,STOP,rev);
move_mic(mic2,STOP,fwd);
move_mic(mic2,STOP,rev);
move_mic(mic3,STOP,fwd);
move_mic(mic3,STOP,rev);
move_mic(mic4,STOP,fwd);
```

```
move_mic(mic4,STOP,rev);
}

void update_cntr(short mic,short dirn,short dist)
{
short temp;

temp = *p40_dinput;

if (dirn) mic_cntr[mic] = mic_cntr_off[mic] + dist - temp;
else      mic_cntr[mic] = mic_cntr_off[mic] - dist + temp;
}

/*********************************************************
   run_mic() moves the specified micrometer "mic" a distance
   "dist" in direction "dirn". Check_cntr() or track_cntr
   should be called before calling run_mic again
*********************************************************/

void run_mic(short mic,short dirn,short dist)
{
/*
if(dirn)
   printf("moving mic%1d forward %6.2f microns\n",mic,((float)dist)/10);
else
   printf("moving mic%1d reverse %6.2f microns\n",mic,((float)dist)/10);
*/
if (dist != 0)
  {
  MicMoving[mic] = TRUE;
  control_word &= disable;
  *p40_doutput = control_word;

 /*  load counter   */

  control_word = (dist>>2) | ld_cnt_mode;
  *p40_doutput = control_word;
  control_word |= enable;
  *p40_doutput = control_word;
  control_word &= disable;
  *p40_doutput = control_word;
  control_word = cntrl_latch | ld_cntr;
 *p40_doutput = control_word;
 control_word |= enable;
 *p40_doutput = control_word;
 control_word |= start_cclk;
 *p40_doutput = control_word;
 control_word &= end_cclk;
 *p40_doutput = control_word;
 control_word &= enable_cntr;
 *p40_doutput = control_word;
 control_word &= disable;
 *p40_doutput = control_word;

 control_word = cntrl_latch | demux_mode;
 *p40_doutput = control_word;
 control_word |= enable;
 *p40_doutput = control_word;

/*  set the desired micrometer and set counter direction    */

 switch (mic)
   {
   case 1:
    if (dirn) control_word = control_word | r1_fwd;
    else control_word = control_word | r1_rev;
    *p40_doutput = control_word;
    break;

   case 2:
    if (dirn) control_word = control_word | r2_fwd;
    else    control_word = control_word | r2_rev;
    *p40_doutput = control_word;
    break;

   case 3:
    if (dirn) control_word = control_word | r3_fwd;
    else    control_word = control_word | r3_rev;
    *p40_doutput = control_word;
    break;

   case 4:
    if (dirn) control_word = control_word | r4_fwd;
    else control_word = control_word | r4_rev;
```

```
    *p40_doutput = control_word;
    break;
   }

/*  set the data flip-flop  */

 control_word |= set_ff;
 *p40_doutput = control_word;
 control_word |= enable;
 *p40_doutput = control_word;
 control_word |= start_dclk;
 *p40_doutput = control_word;
 control_word &= end_dclk;
 *p40_doutput = control_word;
 control_word &= reset_ff;
 *p40_doutput = control_word;
 control_word &= disable;
 *p40_doutput = control_word;
 }
}

void BackComp(short mic,short newdirn)
{
short oldcnt,newcnt;
BackCompDone = 0;

if (newdirn != *(OldMicDirn+mic))
  {
  run_mic(mic,newdirn,(short)(*(MicBacklash+mic)*10.0));
  while(!(quit | no_syncflag) & !BackCompDone)
   {
   oldcnt = *p40_dinput;
   while(pauseflag)
     chkmsg();
   read_data();
   update_timer();
   period = runtime - oldtime;
   oldtime = runtime;
   for (i=0;i<last_chan;i++)
     *(volts+i) = *(volts+i) * *(chan_gain+i);
   for (i=0;i<last_chan;i++)
     *(volts_avgs+i) += *(volts+i);
  calcerrors();
  samp_cntr++;
  filter_all();
  PwrControl();
  if(samp_cntr >= samp_rate)
   {
   samp_cntr = 0;
   for (i=0;i<last_chan;i++)
     *(volts_avgs+i) /= samp_rate;
   store_data();
   for (i=0;i<last_chan;i++)
     *(volts_avgs+i) = 0;
   update_display();
   }
   /*
  if ((runtime - updatetime) >= 0.5)
   {
   updatetime = runtime;
   printf(" Ts = %6.2f ms pwr1 = %4.2f kW pwr2 = %4.2f kW\r",
        1000*period,*(filtvolts+0),sumtotal);
   fflush(stdout);
   }
   */
  chkmsg();
  newcnt = *p40_dinput;
  if (newcnt == oldcnt)
   {
   BackCompDone = TRUE;
   *(MicMoving+mic) = FALSE;
   }
  }
 *(OldMicDim+mic) = newdim;
 }
}


extern struct RastPort *MyRPort;
#define red (short) 5
#define green (short) 4

void DrawDot(int x,int y,short on_off)
{
```

```
if (on_off != GO)
  {
  line(MyRPort,green,(short)(x-2),(short)(y-2),(short)(x+2),(short)(y-2));
  line(MyRPort,green,(short)(x+2),(short)(y-2),(short)(x+2),(short)(y+2));
  line(MyRPort,green,(short)(x+2),(short)(y+2),(short)(x-2),(short)(y+2));
  line(MyRPort,green,(short)(x-2),(short)(y+2),(short)(x-2),(short)(y-2));
  }
else
  {
  line(MyRPort,red,(short)(x-2),(short)(y-2),(short)(x+2),(short)(y-2));
  line(MyRPort,red,(short)(x+2),(short)(y-2),(short)(x+2),(short)(y+2));
  line(MyRPort,red,(short)(x+2),(short)(y+2),(short)(x-2),(short)(y+2));
  line(MyRPort,red,(short)(x-2),(short)(y+2),(short)(x-2),(short)(y-2));
  }
 }
```

DISPLAY.H

```
/*****************************************************************************
display.h contains global variables and defines for the control of data display
*****************************************************************************/

#define width (short)640   /* width and height of display WINDOW */
#define height (short)400

#define initx (short)50    /* width and height of graph  */
#define finalx (short)550
#define lowy (short)335
#define highy (short)65

#define scrollpt (short)400  /* controls how much of old data to scroll */

#define INTUITION_REV 0
#define GRAPHICS_REV 0

#define darkblue (short)0   /* define colors for convenience  */
#define black (short)2
#define white (short)1
#define blue (short)3
#define green (short)4
#define red (short)5
#define cyan (short)6
#define magenta (short)7
#define yellow (short)8
#define orange (short)9

#define graph_color white      /* colors used for drawing graph */
#define graph_back_color black

#define OUTLINEMODE 0

#define gadgHposn 5
#define gadgHspacing 30
#define gadgVposn 3
#define gadgetwidth 20
```

```
#define gadgetheight 13

/*************************************************************
set the horizontal and vertical positions of all the gadgets
*************************************************************/

#define PAUSEH width-75
#define PAUTOH width-75
#define M1AUTOH width-75
#define M2AUTOH width-75
#define FILTH width-75
#define DATAH width-75
#define PARSH width-75
#define PIVH width-75
#define SUMH width-75
#define UNYH gadgHposn
#define UNXH gadgHposn+40
#define POYH gadgHposn+80
#define POXH gadgHposn+120
#define PspH gadgHposn+160
#define UpwrH gadgHposn+200
#define DataTimeX width-75
#define RunTimeX width-75
#define PwrX width-85
#define CurX width-85
#define VolX width-85

#define PAUSEV gadgVposn
#define PAUTOV gadgVposn+20
#define M1AUTOV gadgVposn+40
#define M2AUTOV gadgVposn+60
#define FILTV gadgVposn+80
#define DATAV gadgVposn+100
#define PARSV gadgVposn+120
#define PIVV gadgVposn+140
#define SUMV gadgVposn+160
#define UNXV gadgVposn+gadgetheight+7
#define UNYV gadgVposn+gadgetheight+7
#define POXV gadgVposn+gadgetheight+7
#define POYV gadgVposn+gadgetheight+7
#define PspV gadgVposn+gadgetheight+7
#define UpwrV gadgVposn+gadgetheight+7
#define DataTimeY gadgVposn+300
```

```
#define RunTimeY gadgVposn+265
#define PwrY gadgVposn+190
#define CurY gadgVposn+200
#define VolY gadgVposn+210


struct IntuitionBase *IntuitionBase;
struct GfxBase *GfxBase;
struct IntuiMessage *message;

/************************************************************
   Global variable declarations
************************************************************/
int ScreenFlag;  /* Flag to indicate it is ok to draw to screen */
struct NewScreen NewScreen =
 {
 0,
 0,
 640,
 400,
 4,
 darkblue,white,
 HIRES | LACE,
 CUSTOMSCREEN,
 NULL,
 (UBYTE*)"PLOTTING DISPLAY SCREEN",
 NULL,
 NULL,
 };

/*  Color map for graphing data  */

UWORD colcmap[16] = /* colorized color map */
 {
   0x008, 0xfff, 0x000, 0x00f, 0x0f0, 0xf00, 0x0ff, 0xf0f,
   0xff0, 0xf80, 0x000, 0x00f, 0x0f0, 0xf00, 0x0ff, 0xf0f
 };

short chan_cols[16] =
 {
 red,       /* color for channel 0 */
 green,     /* color for channel 1 */
 yellow,    /* color for channel 2 */
 orange,    /* color for channel 3 */
 red,       /* color for channel 4 */
 red,       /* color for channel 5 */
 red,       /* color for channel 6 */
 green,     /* color for channel 7 */
 green,     /* color for channel 8 */
 green,     /* color for channel 9 */
 yellow,    /* color for channel 10 */
 yellow,    /* color for channel 11 */
 yellow,    /* color for channel 12 */
 orange,    /* color for channel 13 */
 orange,    /* color for channel 14 */
 orange,    /* color for channel 15 */
 };
#define UNYCol  (short)red
#define UNXCol  (short)green
#define POYCol  (short)cyan
#define POXCol  (short)magenta
#define PspCol (short)orange
#define UpwrCol (short)green


/*  define the outline of the graph  */

short graphoutline[10] =
{
initx,highy,
finalx,highy,
finalx,lowy,
initx,lowy,
initx,highy
};

/*  structures used in display  */

struct RastPort *MyRPort;
struct Screen *Screen1;
struct NewWindow NewWindow;
struct Window *DispWind;
struct BitMap bmap;
```

```
short midpt;                /* vertical midpt of graph      */
short graphx;                /* current x graph point        */
short graphy;                /* current y graph point        */
short old_x;                /* last x graph point          */
short oldy;                /* last y graph point          */
short oldtotal;
short graph_chans[16];       /* channels to plot  */
short UNXFlag,UNYFlag;  /* plot flag for derived unif signals */
short POXFlag,POYFlag;  /* plot flags for derived posn signals */
short PapFlag;     /* plot flag for power setpoint */
short sumflag;   /* plot flag for sum of sensor signals */
short pivflag;   /* indicates choice of first four data channels */
short filtflag;  /* indicates whether data is being digitally filtered */
short UpwrFlag;  /* indicates whether power control signal is plotted */
short old_y[16];
short *scrollbuf;
short scrollcnt;
double yscale;
double xscale;
double maxtime,mintime;
short NewMaxTime;
char TextBuf[20];
short numdigs;
short pt_spacing;
short numticks;
char* strbuf;
short pt_spacing;

/*  variables used in digital filter routines     */
float xbuf[30][4];
float ybuf[30][4];
short n = 3;
short nm1 = 2;
short nm2 = 1;
short nm3 = 0;
float fc;            /* cutoff frequency */

short testnum = 1;
short numtests = 4;

char *gfxtime;
char *buf;
```

```
/**********************************************************************
   structures for analog channel selection gadgets
   (initialized at runtime)
**********************************************************************/

struct IntuiText chanText[16];
struct Gadget chanGadget[16];
int GadgetPosn[20];

/**********************************************************************
   Structure assignments for remaining gadgets
   NOTE: Gadget structures are initialized at
   runtime [see init_display()]
**********************************************************************/


struct IntuiText parsText =
{
(UBYTE)white,(UBYTE)black,      /*  FrontPen,BackPen */
(UBYTE)JAM2,                   /*  DrawMode  */
(SHORT)2,                    /*  LeftEdge  */
(SHORT)3,                    /*  TopEdge   */
NULL,                      /*  Pointer to ITextFont  */
(UBYTE*)"PARS",                /*  Pointer to IText  */
NULL                       /*  Pointer to NextText  */
};

struct Gadget PARSGadget =
{
NULL,                      /*  Pointer to Next Gadget  */
PARSH,PARSV,                  /*  LeftEdge,TopEdge */
43,gadgetheight,              /*  Width, Height  */
GADGHBOX,                     /*  Flags   */
TOGGLESELECT | GADGIMMEDIATE,    /*  Activation */
BOOLGADGET,                   /*  GadgetType   */
NULL,                      /*  pointer to GadgetRender  */
NULL,                      /*  pointer to SelectRender  */
&parsText,                   /*  pointer to GadgetText  */
0L,                       /*  no MutualExclude  */
NULL,                      /*  pointer to SpecialInfo  */
NULL,                      /*  pointer to UserData   */
NULL
};
```

```
struct IntuiText UpwrText =
{
(UBYTE)UpwrCol,(UBYTE)black,     /* FrontPen,BackPen */
(UBYTE)JAM2,                 /* DrawMode */
(SHORT)2,                  /* LeftEdge */
(SHORT)3,                  /* TopEdge */
NULL,                    /* Pointer to ITextFont */
(UBYTE*)"Upwr",              /* Pointer to IText */
NULL                     /* Pointer to NextText */
};

struct Gadget UpwrGadget =
{
&PARSGadget,                 /* Pointer to Next Gadget */
UpwrH,UpwrV,                 /* LeftEdge,TopEdge */
36,gadgetheight,            /* Width, Height */
GADGHBOX,                   /* Flags */
TOGGLESELECT | GADGIMMEDIATE,    /* Activation */
BOOLGADGET,                  /* GadgetType */
NULL,                    /* pointer to GadgetRender */
NULL,                    /* pointer to SelectRender */
&UpwrText,                  /* pointer to GadgetText */
0L,                     /* no MutualExclude */
NULL,                    /* pointer to SpecialInfo */
NULL,                    /* pointer to UserData */
NULL
};

struct IntuiText PspText =
{
(UBYTE)PspCol,(UBYTE)black,      /* FrontPen,BackPen */
(UBYTE)JAM2,                 /* DrawMode */
(SHORT)2,                  /* LeftEdge */
(SHORT)3,                  /* TopEdge */
NULL,                    /* Pointer to ITextFont */
(UBYTE*)"Psp",               /* Pointer to IText */
NULL                     /* Pointer to NextText */
};

struct Gadget PspGadget =
{
&UpwrGadget,                 /* Pointer to Next Gadget */
PspH,PspV,                  /* LeftEdge,TopEdge */
30,gadgetheight,             /* Width, Height */
GADGHBOX,                   /* Flags */
TOGGLESELECT | GADGIMMEDIATE,    /* Activation */
BOOLGADGET,                  /* GadgetType */
NULL,                    /* pointer to GadgetRender */
NULL,                    /* pointer to SelectRender */
&PspText,                   /* pointer to GadgetText */
0L,                     /* no MutualExclude */
NULL,                    /* pointer to SpecialInfo */
NULL,                    /* pointer to UserData */
NULL
};

struct IntuiText POYText =
{
(UBYTE)POYCol,(UBYTE)black,      /* FrontPen,BackPen */
(UBYTE)JAM2,                 /* DrawMode */
(SHORT)2,                  /* LeftEdge */
(SHORT)3,                  /* TopEdge */
NULL,                    /* Pointer to ITextFont */
(UBYTE*)"POY",               /* Pointer to IText */
NULL                     /* Pointer to NextText */
};

struct Gadget POYGadget =
{
&PspGadget,                  /* Pointer to Next Gadget */
POYH,POYV,                  /* LeftEdge,TopEdge */
30,gadgetheight,             /* Width, Height */
GADGHBOX,                   /* Flags */
TOGGLESELECT | GADGIMMEDIATE,    /* Activation */
BOOLGADGET,                  /* GadgetType */
NULL,                    /* pointer to GadgetRender */
NULL,                    /* pointer to SelectRender */
&POYText,                   /* pointer to GadgetText */
0L,                     /* no MutualExclude */
NULL,                    /* pointer to SpecialInfo */
NULL,                    /* pointer to UserData */
NULL
```

```
};

struct IntuiText POXText =
{
(UBYTE)POXCol,(UBYTE)black,     /* FrontPen,BackPen */
(UBYTE)JAM2,                    /* DrawMode */
(SHORT)2,                       /* LeftEdge */
(SHORT)3,                       /* TopEdge  */
NULL,                           /* Pointer to ITextFont */
(UBYTE*)"POX",                  /* Pointer to IText */
NULL                            /* Pointer to NextText */
};

struct Gadget POXGadget =
{
&POYGadget,                     /* Pointer to Next Gadget */
POXH,POXV,                      /* LeftEdge,TopEdge */
30,gadgetheight,                /* Width, Height */
GADGHBOX,                       /* Flags */
TOGGLESELECT | GADGIMMEDIATE,   /* Activation */
BOOLGADGET                      /* GadgetType */
NULL,                           /* pointer to GadgetRender */
NULL,                           /* pointer to SelectRender */
&POXText,                       /* pointer to GadgetText */
0L,                             /* no MutualExclude */
NULL,                           /* pointer to SpecialInfo */
NULL,                           /* pointer to UserData */
NULL
};

struct IntuiText UNYText =
{
(UBYTE)UNYCol,(UBYTE)black,     /* FrontPen,BackPen */
(UBYTE)JAM2,                    /* DrawMode */
(SHORT)2,                       /* LeftEdge */
(SHORT)3,                       /* TopEdge  */
NULL,                           /* Pointer to ITextFont */
(UBYTE*)"UNY",                  /* Pointer to IText */
NULL                            /* Pointer to NextText */
};

struct Gadget UNYGadget =
{
&POXGadget,                     /* Pointer to Next Gadget */
UNYH,UNYV,                      /* LeftEdge,TopEdge */
30,gadgetheight,                /* Width, Height */
GADGHBOX,                       /* Flags */
TOGGLESELECT | GADGIMMEDIATE,   /* Activation */
BOOLGADGET,                     /* GadgetType */
NULL,                           /* pointer to GadgetRender */
NULL,                           /* pointer to SelectRender */
&UNYText,                       /* pointer to GadgetText */
0L,                             /* no MutualExclude */
NULL,                           /* pointer to SpecialInfo */
NULL,                           /* pointer to UserData */
NULL
};

struct IntuiText UNXText =
{
(UBYTE)UNXCol,(UBYTE)black,     /* FrontPen,BackPen */
(UBYTE)JAM2,                    /* DrawMode */
(SHORT)2,                       /* LeftEdge */
(SHORT)3,                       /* TopEdge  */
NULL,                           /* Pointer to ITextFont */
(UBYTE*)"UNX",                  /* Pointer to IText */
NULL                            /* Pointer to NextText */
};

struct Gadget UNXGadget =
{
&UNYGadget,                     /* Pointer to Next Gadget */
UNXH,UNXV,                      /* LeftEdge,TopEdge */
30,gadgetheight,                /* Width, Height */
GADGHBOX,                       /* Flags */
TOGGLESELECT | GADGIMMEDIATE,   /* Activation */
BOOLGADGET,                     /* GadgetType */
NULL,                           /* pointer to GadgetRender */
NULL,                           /* pointer to SelectRender */
&UNXText,                       /* pointer to GadgetText */
0L,                             /* no MutualExclude */
NULL,                           /* pointer to SpecialInfo */
NULL,                           /* pointer to UserData */
NULL
```

```
};

struct IntuiText PIVText =
{
(UBYTE)white,(UBYTE)black,      /* FrontPen,BackPen */
(UBYTE)JAM2,                /* DrawMode */
(SHORT)2,                /* LeftEdge */
(SHORT)3,                /* TopEdge  */
NULL,                /* Pointer to ITextFont */
(UBYTE*)"PIV",                /* Pointer to IText */
NULL                /* Pointer to NextText */
};

struct Gadget PIVGadget =
{
&UNXGadget,                /* Pointer to Next Gadget */
PIVH,PIVV,                /* LeftEdge,TopEdge */
28,gadgetheight,                /* Width, Height */
GADGHBOX | SELECTED,                /* Flags */
TOGGLESELECT | GADGIMMEDIATE,    /* Activation */
BOOLGADGET,                /* GadgetType  */
NULL,                /* pointer to GadgetRender */
NULL,                /* pointer to SelectRender */
&PIVText,                /* pointer to GadgetText */
0L,                /* no MutualExclude */
NULL,                /* pointer to SpecialInfo */
NULL,                /* pointer to UserData  */
NULL
};

struct IntuiText PAUTOText =
{
(UBYTE)white,(UBYTE)black,      /* FrontPen,BackPen */
(UBYTE)JAM2,                /* DrawMode */
(SHORT)2,                /* LeftEdge */
(SHORT)3,                /* TopEdge  */
NULL,                /* Pointer to ITextFont */
(UBYTE*)"PAUTO",                /* Pointer to IText */
NULL                /* Pointer to NextText */
};

struct Gadget PAUTOGadget =
```

```
{
&PIVGadget,                /* Pointer to Next Gadget */
PAUTOH,PAUTOV,                /* LeftEdge,TopEdge */
43,gadgetheight,                /* Width, Height */
GADGHBOX,                /* Flags */
TOGGLESELECT | GADGIMMEDIATE,    /* Activation */
BOOLGADGET,                /* GadgetType */
NULL,                /* pointer to GadgetRender */
NULL,                /* pointer to SelectRender */
&PAUTOText,                /* pointer to GadgetText */
0L,                /* no MutualExclude */
NULL,                /* pointer to SpecialInfo */
NULL,                /* pointer to UserData  */
NULL
};

struct IntuiText DataText =
{
(UBYTE)white,(UBYTE)black,      /* FrontPen,BackPen */
(UBYTE)JAM2,                /* DrawMode */
(SHORT)2,                /* LeftEdge */
(SHORT)3,                /* TopEdge  */
NULL,                /* Pointer to ITextFont */
(UBYTE*)"DATA",                /* Pointer to IText */
NULL                /* Pointer to NextText */
};

struct Gadget DataGadget =
{
&PAUTOGadget,                /* Pointer to Next Gadget */
DATAH,DATAV,      /* LeftEdge,TopEdge */
36,gadgetheight,                /* Width, Height */
GADGHBOX,                /* Flags */
TOGGLESELECT|GADGIMMEDIATE,     /* Activation */
BOOLGADGET,                /* GadgetType */
NULL,                /* pointer to GadgetRender */
NULL,                /* pointer to SelectRender */
&DataText,                /* pointer to GadgetText */
0L,                /* no MutualExclude */
NULL,                /* pointer to SpecialInfo */
NULL,                /* pointer to UserData */
NULL
};

struct IntuiText FiltText =
{
(UBYTE)white,(UBYTE)black,        /* FrontPen,BackPen */
(UBYTE)JAM2,                  /* DrawMode */
(SHORT)2,                  /* LeftEdge */
(SHORT)3,                  /* TopEdge */
NULL,                  /* Pointer to ITextFont */
(UBYTE*)"FILTER",             /* Pointer to IText */
NULL                   /* Pointer to NextText */
};

struct Gadget FiltGadget =
{
&DataGadget,                        /* Pointer to Next Gadget */
FILTH,FILTV,         /* LeftEdge,TopEdge */
52,gadgetheight,               /* Width, Height */
GADGHBOX,                       /* Flags */
TOGGLESELECT | GADGIMMEDIATE,    /* Activation */
BOOLGADGET,                      /* GadgetType */
NULL,                    /* pointer to GadgetRender */
NULL,                    /* pointer to SelectRender */
&FiltText,               /* pointer to GadgetText */
0L,                    /* no MutualExclude */
NULL,                    /* pointer to SpecialInfo */
NULL,                    /* pointer to UserData */
NULL
};

struct IntuiText SumText =
{
(UBYTE)white,(UBYTE)black,        /* FrontPen,BackPen */
(UBYTE)JAM2,                  /* DrawMode */
(SHORT)2,                  /* LeftEdge */
(SHORT)3,                  /* TopEdge */
NULL,                  /* Pointer to ITextFont */
(UBYTE*)"SUM",            /* Pointer to IText */
NULL                   /* Pointer to NextText */
};

struct Gadget SumGadget =
```

```
{
&FiltGadget,                        /* Pointer to Next Gadget */
SUMH,SUMV,         /* LeftEdge,TopEdge */
28,gadgetheight,               /* Width, Height */
GADGHBOX,                       /* Flags */
TOGGLESELECT | GADGIMMEDIATE,    /* Activation */
BOOLGADGET,                      /* GadgetType */
NULL,                    /* pointer to GadgetRender */
NULL,                    /* pointer to SelectRender */
&SumText,                /* pointer to GadgetText */
0L,                    /* no MutualExclude */
NULL,                    /* pointer to SpecialInfo */
NULL,                    /* pointer to UserData */
NULL
};

struct IntuiText M1AUTOText =
{
(UBYTE)white,(UBYTE)black,        /* FrontPen,BackPen */
(UBYTE)JAM2,                  /* DrawMode */
(SHORT)2,                  /* LeftEdge */
(SHORT)3,                  /* TopEdge */
NULL,                  /* Pointer to ITextFont */
(UBYTE*)"PRIM",             /* Pointer to IText */
NULL                   /* Pointer to NextText */
};

struct Gadget M1AUTOGadget =
{
&SumGadget,                         /* Pointer to Next Gadget */
M1AUTOH,M1AUTOV,       /* LeftEdge,TopEdge */
36,gadgetheight,               /* Width, Height */
GADGHBOX,                       /* Flags */
TOGGLESELECT | GADGIMMEDIATE,    /* Activation */
BOOLGADGET,                      /* GadgetType */
NULL,                    /* pointer to GadgetRender */
NULL,                    /* pointer to SelectRender */
&M1AUTOText,             /* pointer to GadgetText */
0L,                    /* no MutualExclude */
NULL,                    /* pointer to SpecialInfo */
NULL,                    /* pointer to UserData */
NULL
};


struct IntuiText M2AUTOText =
{
(UBYTE)white,(UBYTE)black,      /* FrontPen,BackPen */
(UBYTE)JAM2,                    /* DrawMode */
(SHORT)2,                       /* LeftEdge */
(SHORT)3,                       /* TopEdge */
NULL,                       /* Pointer to ITextFont */
(UBYTE*)"FDBK",                 /* Pointer to IText */
NULL                        /* Pointer to NextText */
};

struct Gadget M2AUTOGadget =
{
&M1AUTOGadget,                          /* Pointer to Next Gadget */
M2AUTOH,M2AUTOV,        /* LeftEdge,TopEdge */
36,gadgetheight,                /* Width, Height */
GADGHBOX,                       /* Flags */
TOGGLESELECT | GADGIMMEDIATE,   /* Activation */
BOOLGADGET,                       /* GadgetType */
NULL,                       /* pointer to GadgetRender */
NULL,                       /* pointer to SelectRender */
&M2AUTOText,                    /* pointer to GadgetText */
0L,                       /* no MutualExclude */
NULL,                       /* pointer to SpecialInfo */
NULL,                       /* pointer to UserData */
NULL
};

struct IntuiText pauseText =
{
(UBYTE)white,(UBYTE)black,      /* FrontPen,BackPen */
(UBYTE)JAM2,                    /* DrawMode */
(SHORT)2,                       /* LeftEdge */
(SHORT)3,                       /* TopEdge */
NULL,                       /* Pointer to ITextFont */
(UBYTE*)"PAUSE",                /* Pointer to IText */
NULL                        /* Pointer to NextText */
};
```

```
struct Gadget pauseGadget =
{
&M2AUTOGadget,                /* Pointer to Next Gadget */
PAUSEH,PAUSEV,              /* LeftEdge,TopEdge */
45,gadgetheight,            /* Width, Height */
GADGHBOX | SELECTED,             /* Flags */
TOGGLESELECT | GADGIMMEDIATE,    /* Activation */
BOOLGADGET,                   /* GadgetType */
NULL,                   /* pointer to GadgetRender */
NULL,                   /* pointer to SelectRender */
&pauseText,                /* pointer to GadgetText */
0L,                   /* no MutualExclude */
NULL,                   /* pointer to SpecialInfo */
NULL,                   /* pointer to UserData */
NULL
};
```

P40.C

```
/*************************************************************************
THIS IS A MODIFIED VERSION OF THE ORIGINAL PROTO-40K SOURCE
CODE.  UNNEEDED ROUTINES HAVE BEEN DELETED. IF OTHER ROUTINES
ARE DESIRED, RETURN TO THE ORIGINAL P40.C FILE IN THE PROTO-40K
DIRECTORY
**************************************************************************/

/*************************************************************
* p40.c V1.1                    6/29/89 *
**************************************************************
*  ACDA Proto-40k Rev B. Suggested Interface code.        *
**************************************************************
*                        *
* 16 A/D channels (0-10 V) or (-10 - +10 V) input range .    *
*    Programmable gain input amplifier                  *
*  2 D/A channels (0-14 V swing)                        *
*  3 Fully programmable timer channels                      *
* 16 digital Input bits                     *
* 16 digiatl Output bits                       *
*                        *
**************************************************************
*                        *
* (C) Copyright 1989 ACDA Corporation.  All rights reserved.     *
* Permission is given to use this code in commercial/non-commercial  *
* applications as long as this notice is left in place.      *
*                        *
**************************************************************
*                        *
*  Suggested Interface Functions:                    *
*                        *
*  p40_init    - Initialize the p40 for use.               *
*                        *
* short p40_init()                          *
*  If it succeeds it will return 1.  The card will auto-       *
*  configure in the hardware configuration tables and the       *
*  Proto-40k address map variables will be initialized.       *
*
**************************************************************/

#include <exec/types.h>
#include <libraries/expansion.h>
#include <libraries/configregs.h>
#include <libraries/configvars.h>
#include <functions.h>
#include <math.h>

#include "p40.h"

/* Proto-40k Registers  ********************************************/
short *p40_data;    /* data area                   */
short *p40_mux;     /* A/D Multiplexer (MUX) address       */
short *p40_gain;    /* for programmable gain option (set 0,1,2,3)  */
short *p40_strobe;  /* read or write to start conversion        */
short *p40_end;     /* read for end of conversion            */
short *p40_da1;     /* write byte for D/A, last val stays latched  */
short *p40_da2;     /* write byte for D/A, last val stays latched  */
short *p40_dinput;  /* read digital input bits          */
short *p40_doutput; /* write digital output bits           */
short *p40_modeselect;  /* "random"=0, "sequential"=2 A/D addressing   */
short *p40_time0;   /* 8254 timer channel 0 (139 nsec resolution   */
short *p40_time1;   /* 8254 timer channel 1 (278 nsec resolution   */
short *p40_time2;   /* 8254 timer channel 2 (139 nsec resolution   */
short *p40_timecontrol; /* 8254 timer control register          */
/*****************************************************************/

short controlbits[16] = {0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0};

short *config();

short p40_init(void)
{
   short *base;

   if ((base = config(ACDA_CORPORATION, ACDA_PROTO40K, 0L)) == 0)
      return(0);
   /*
      Addressing format for the Proto-40k
      ---------------------------------
     Register    =     Auto-config base + Offset
*/

  p40_time0  = (short *) ((long) base + 0x0000);
  p40_time1  = (short *) ((long) base + 0x0004);
  p40_time2  = (short *) ((long) base + 0x0002);
  p40_timecontrol = (short *) ((long) base + 0x0006);
  p40_data   = (short *) ((long) base + 0x1000);
  p40_mux    = (short *) ((long) base + 0x2000);
  p40_gain   = (short *) ((long) base + 0x3000);
  p40_strobe  = (short *) ((long) base + 0x4000);
  p40_end    = (short *) ((long) base + 0x5000);
  p40_da1    = (short *) ((long) base + 0x7000);
  p40_da2    = (short *) ((long) base + 0x7002);
  p40_dinput  = (short *) ((long) base + 0x8000);
  p40_doutput = (short *) ((long) base + 0x8002);
  p40_modeselect  = (short *) ((long) base + 0x9000);
  return(1);
}


/*****************************************************************
* Proto-40k Auto-Configuration Code:                            *
*****************************************************************
* config() will find the Proto-40k or other expansion device and *
* assign its base address.                                       *
*****************************************************************
*  Note:                                                        *
*  This code and our circuitry is used with                      *
*  KickStart/WorkBench V1.2 configuration scheme.               *
*****************************************************************/

struct  ConfigDev   *FindConfigDev();
void    *OpenLibrary();
struct  ExpansionBase   *ExpansionBase;


short *config(long manu, long prod, long unit)
{
  struct ConfigDev *board;

  ExpansionBase = OpenLibrary(EXPANSIONNAME,0L);
  if (ExpansionBase == 0) {
    printf("Could not open Expansion Library!\n");
    return ((short *) 0);
  }
  board = 0L;
  if((board = (struct ConfigDev *)FindConfigDev(NULL,
          manu, prod))==0) {
    return((short *) 0);
  }
  return((short *) board->cd_BoardAddr);
}


/*****************************************************************
* Amiga System Timer Functions:                                 *
*****************************************************************
* The following Amiga system timer functions are provided as a    *
* convenience to programmers.  These timer functions may be used to *
* cross-check requested sample-rates, calculate elapsed times, etc.  *
* BEWARE! They are only as accurate as the Amiga system software  *
* permits.                                                      *
*****************************************************************/

#include <devices/timer.h>

struct MsgPort *CreatePort();
long   TimerBase;
struct timeval time1, time2;
struct timerequest t;

short Open_systimer(void)
{
  if (OpenDevice(TIMERNAME, UNIT_MICROHZ, &t.tr_node, 0L)) {
    printf("Timer is not available.\n");
    return(0);
  }
  t.tr_node.io_Message.mn_ReplyPort = CreatePort(0L, 0L);
  return(1);
}

void Close_systimer(void)
```

```
{
   CloseDevice(&t.tr_node);
   DeletePort(t.tr_node.io_Message.mn_ReplyPort);
}

void get_time1(struct timeval *time)    /* t1 marker for calculation of
                                           elapsed time */
{
   t.tr_node.io_Command = TR_GETSYSTIME;
   DoIO(&t.tr_node);
   time->tv_secs = t.tr_time.tv_secs;
   time->tv_micro = t.tr_time.tv_micro;
}

short p40_dtoa(short chan, double val)
{
   switch(chan) {
     case  0: *p40_da1 = val * 255/15;    break;
     case  1: *p40_da2 = val * 255/15;    break;
   }
}
```

P40.H

```
/* p40.h */

#define ACDA_CORPORATION   2070L     /* Manufacturer number */
#define ACDA_PROTO40K      4L     /* Product number */

#define HEX          0     /* A/D Output format */
#define DECIMAL        1
#define VOLTS         2

#define RANDOM        0     /* A/D addressing modes */
#define SEQUENTIAL     1

#define BITMASK(n)     (1 << (n))
#define TIMER_RATE     (139*2)    /* 8254 channel 2 (nanosecs) */
#define TIMER_FAST     139    /* 8254 channel 1,3(nanosecs) */

#define CONVERT_AD_VOLTS(j) ((float)(j & 0x0fff)/4095.0 * 10.0)
#define CONVERT_AD_VOLTSBI2C(j) ((float)(((j<<4)>>4)/4096.0*20.0))
#define CONVERT_AD_VOLTSBIOF(j) ((float)(j & 0x0fff -2048)/2048.0 * 10.0))
#define CONVERT_DA_VOLTS(j) ((float)(j)/255.0 * 15.0)

extern short controlbits[16];  /* 8254 timer array of control bits      */
extern short *p40_time0;   /* 8254 timer channel 0 (139 nsec resolution) */
extern short *p40_time1;   /* 8254 timer channel 1 (278 nsec resolution) */
extern short *p40_time2;   /* 8254 timer channel 2 (139 nsec resolution) */
extern short *p40_timecontrol;  /* 8254 timer control register */
extern short *p40_data;    /* data area */
extern short *p40_mux;     /* mux address */
extern short *p40_gain;    /* for programmable gain option (set 0,1,2,3 */
extern short *p40_strobe;  /* read or write to start conversion */
extern short *p40_end;     /* read for end of conversion */
extern short *p40_da1;     /* write byte for D/A, last val stays latched */
extern short *p40_da2;     /* write byte for D/A, last val stays latched */
extern short *p40_dinput;  /* digital input word */
extern short *p40_doutput;  /* digital output word */
extern short *p40_modeselect; /* "random"=0, "sequential"=2 A/D addressing */
```

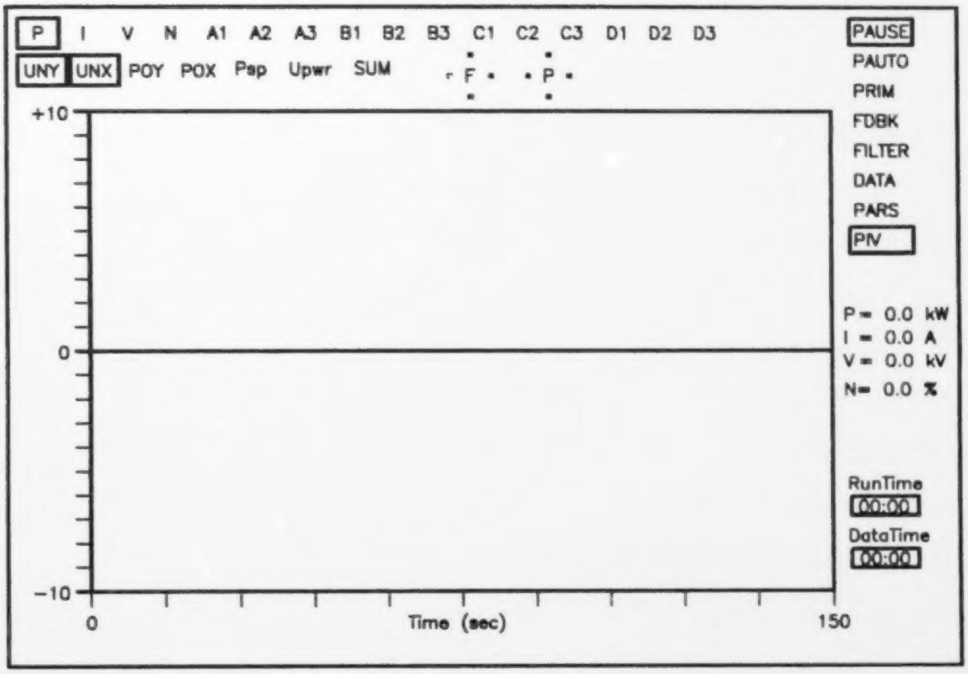
```
struct DASstruct {      /* Desribes a data buffer       */
   short channel;      /* which channel            */
   unsigned long  type;    /* memory type CHIP or FAST         */
   unsigned short rate;    /* user requested sample rate in Hertz  */
   unsigned long  length;  /* requested rate x secs for each buffer*/
   float secs;     /* recording period duration in secs    */
   short gain;     /* gain set = 0,1,2,3 (x1,x10,x100,x500)*/
   short mode;     /* Random or sequential read mode   */
   char  name[32];     /* individual's buffer name         */
   short *buf;     /* location of memory to get/put data   */
};

extern struct DASstruct A2DBuf[16]; /* Suggested channel structures     */
extern struct DASstruct D2ABuf[2];
extern struct DASstruct DIBuf[16];
extern struct DASstruct DOBuf[16];

            /*
              For Rev 0 boards, must use reverse_bits() to
              tranform reversed offset binary to binary
              A/D data.
            */

#define reverse_bits(bit)   (nibble_table[((bit) & 0x000f) >> 0] << 8) | \
            (nibble_table[((bit) & 0x00f0) >> 4] << 4) | \
            (nibble_table[((bit) & 0x0f00) >> 8] << 0)

extern short readchannel();
extern char nibble_table[];

extern short p40_init();
extern short p40_dtoa(short chan, double val);
extern short p40_atod();
extern short p40_mask();
extern short Open_systimer();       /* system timer functions */
extern float get_time2();
```

MAKEFILE

```
OBJ1 = p40.o subs.o control.o

CFLAGS = -mcd -c2 -f8 -wdao

control: $(OBJ1)
   ln $(OBJ1) -o $@ -lm8l -lcl

p40.o: p40.h
subs.o: p40.h control.h display.h
control.o: p40.h control.h display.h
```

APPENDIX V - OPERATING MANUAL FOR CONTROL/DATA-ACQUISITION SYSTEM

This appendix contains additional detail concerning the operation of the data-acquisition and control system described in this thesis. For the most part, operation of the system is quite intuitive. However, the custom designed nature of the system dictates that some sort of reference manual be provided.



P= 0.0 kW
I = 0.0 A
V= 0.0 kV
N= 0.0 %

RunTime
00:00
DataTime
00:00

Fig. 1 Graphical User Interface Screen

I. Graphical Interface Operation

To run the program, open a CLI shell from the AMIGA'S workdisk and change to the "PI4:CONTROL" directory . From here the program may be loaded using the command line "RUN CONTROL". The program can also be loaded by simply entering "CONTROL", however, by using the RUN command the CLI shell is left free for other tasks while the control program is running.

The user is presented with the interface screen shown in Fig. 1. Along the top of the display are a number of mouse gadgets which are used to select which of the 16 available raw data sources are currently plotted. These data sources are described below, beginning at the left:

P	laser output power(kW)
I	laser discharge current (A) (displayed with a 30 amp offset to allow display inside the ± 10 plotting limits)
V	laser discharge voltage (kV)
N	lasing efficiency
A1	Segment ring A1 of the sensor array (see Chapter 5)
A2	Segment ring A2 of the sensor array
A3	Segment ring A3 of the sensor array
B1	Segment ring B1 of the sensor array
B2	Segment ring B2 of the sensor array
B3	Segment ring B3 of the sensor array
C1	Segment ring C1 of the sensor array
C2	Segment ring C2 of the sensor array
C3	Segment ring C3 of the sensor array
D1	Segment ring D1 of the sensor array
D2	Segment ring D2 of the sensor array
D3	Segment ring D3 of the sensor array

The second row of mouse gadgets allows selection of six additional derived signals that are particularly useful for performance monitoring of the mode controller. These are:

UNY	Uniformity error for the vertical direction
UNX	Uniformity error for the horizontal direction
POY	Position error for the vertical direction
POX	Position error for the horizontal direction
Psp	Current power setpoint
Upwr	Current power controller output
SUM	Sum total of all signals scaled to provide a representation of total output power in kW

To the right of these are a pair of indicators which inform the user of alignment servo activity. "F" indicates the feedback (convex) mirror and "P" describes the primary (concave) mirror. The small squares above and below, and to the sides of these letters are highlighted in red to show when a servo is in motion. The directions were selected to correspond with the effect upon the annular beams appearance which results when the various servos are adjusted. Down the right hand side of the plotting area are additional mouse gadgets which control the operation of the program. Beginning at the top the functions are as listed below:

PAUSE-Allows the user to pause the program at any point. Controller operation is suspended and the data display is frozen. If servo movements were currently under way, they are halted immediately.

PAUTO-Enables automatic power control. The PIV mouse gadget (see below) must be selected, and the switch on the SCR current controller must be set to the computer position. Also, this may be used only when the laser output is dumped into the water cone power meter.

PRIM-Enables control of the primary (convex) mirror. This allows control of mode uniformity. The micrometer source selector box must be set to the "COMPUTER" position in order for control signals from the computer to reach the micrometer drivers. The uniformity control switch on the hardware controller must also be set to automatic because the computer simply relays the signals from the electronic controller through to the micrometers to accomplish uniformity control.

FDBK-Enables control of the feedback (concave) mirror. This mouse gadget may only be selected when the "PRIM" mouse gadget (see above) is selected. It is important that the micrometer speeds be adjusted using the "SETSPEED" program before enabling this feature. Speeds of 4.0 μ/sec for all 4 micrometers should provide acceptable results.

FILTER-Enables digital filtering of displayed data. Two types of filters are available. These are a third order Butterworth and a simple first order lowpass. The type currently used is determined by the setting of a variable in the parameter file (CONTROL.PAR) which is described later.

DATA-Enables storage of ASCII data in a RAM data file. The rate at which data is stored may be controlled to some extent (0.1 seconds or greater in multiples of 0.1 second). This is controlled by a variable in the "CONTROL.PAR" parameter file.

PARS-Reads in any changes which have been made to the parameter file. The user may use the multitasking capability of the AMIGA system to perform on-line modifications to the parameter file. After saving the changes to disk, this gadget reinitializes the affected variables.

Below these gadgets, six numerical displays are provided. As can be seen in Fig. 1, laser power, discharge current and voltage are displayed continuously. Laser efficiency is also calculated, based on input and output power, and displayed. Two timers are provided: the first of these continuously displays elapsed time since the program began operation. The timer is paused whenever the PAUSE gadget is selected. The "DataTime" timer simply displays elapsed time since data storage began. It is reset each time the DATA gadget is disabled.

Data Collection

Data collection is convenient when the following precautions are observed. When the DATA gadget is selected the program opens a file on the ram disk (ram:datafile) and begins storing raw data from a number of sources. When the DATA gadget is deselected this file is closed. If the user wishes to save the data he must then enter a CLI window and copy the RAM data file to the hard disk. **If the DATA gadget is again selected before this operation is performed the data will be lost.** In its present configuration the program stores raw data from the 16 analog data channels as well as a number of other signals. A user with C-programming experience may easily change which data is stored by modifying the subroutine "StoreData" in the CONTROL.C program file. (See appendix 4 for software listings)

Parameter Modification

Various parameters such as gain, filter cutoff frequencies, etc., may be modified on-line using a standard text editor. The easiest approach is to use the command line "AZ CONTROL.PARS" before running the control program. Once inside the editor the various functions of the parameters are explained. The new parameters may be read into the program by clicking on the "PARS" gadget.
(Use care not to change the order of the parameters in this file. The routine which reads the parameters into the program is relatively unsophisticated and simply reads the values in a fixed order without checking to see if they correspond to the correct name.)

Motor-Micrometer Speed Adjustment

Correct operation of the alignment controllers requires that the speed of the micrometers be set to the proper values before running the system. Since the motor-micrometer drivers allow manual setting of the speeds in the forward and reverse direction but provide no indication of the actual selected speed, the following procedure was used to accurately adjust the speeds. First, the user enters a CLI window and enters the "PI4:CONTROL" directory. Then run the program "SETSPEED". This program simply jogs each micrometer forward and reverse, measuring the time required to move a distance of 20.0 µm. After each forward and reverse movement the program provides a readout of forward and reverse speed. If the two values differ by more than 1 µm/s, the program prompts the user to adjust the speeds as needed. Although by no means optimum, speeds of 4.0 µm/s were found to provide acceptable performance.

II. Operation of the Hardwired Controller

Alignment of the Sensor Array

The sensor array is mounted on an x-y table which allows adjustment of its vertical and horizontal position. Proper positioning of the array is required to ensure proper operation of the alignment system. This may be necessary if the laser has been worked on, or if the position of the final turning mirror has been altered. The alignment procedure is as follows:

1. Before starting the laser, monitor the x and y uniformity signals. (deselect the PIV gadget and change the switch on the rear of the electronics panel away from the PIV position, then choose channels Ux and Uy on the graphical screen-top row) While monitoring these signals, adjust the small pots on the uniformity controller board to obtain error signals of zero for both directions. Be sure to adjust the offsets and not the gains. (the offset pots are the bottom and second from the top holes for the vertical and horizontal directions respectively)
2. Start the laser (first run the wand and power up the hardwired controller).
3. Manually align the resonator mirrors to obtain the best possible output beam.
4. Adjust the position of the sensor array to return the error signals to zero.
5. Place the micrometer selector switch in the "CONTROL" position and place the uniformity controller on automatic.
6. If necessary, fine tune the centering pots to obtain a uniform beam.

7. With the controller operating, select the UNY and UNX gadgets. These signals are the uniformity signals for the vertical and horizontal directions. (similar to the ones monitored by the Ux and Uy channels but derived from the raw sensor data rather than from the hardwired controller signals) The idea now is to adjust the offsets and gains in the parameter file so that these signals match the ones from the hardwired controller. Adjust the parameters Kcyu, Kcxu, yoffsetu and xoffsetu so that the signals match those from the controller electronics. During this procedure, it may be easier to induce a deliberate misalignment to check how closely the signals match.
8. Return the switch on the rear of the electronics panel back to the PIV position and select PIV on the data-acquistion screen. This allows monitoring of laser power, current and voltage.
9. The uniformity controller may now be used simply by placing the micrometer select switch in the "CONTROL" position and placing the uniformity controller toggle switch in the "AUTO" position. Do not enable the toggle switch on the position controller board. This switch is obsolete.